Crowd-Driven Computer Vision

Kathleen Tuite

A dissertation
submitted in partial fulfillment of the
requirements for the degree of

Doctor of Philosophy

University of Washington

2019

Reading Committee:

Linda Shapiro, Chair

Steve Tanimoto

Daniel Avrahami

Alex Colburn

Program Authorized to Offer Degree:
Computer Science and Engineering

University of Washington

Abstract

Crowd-Driven Computer Vision

Kathleen Tuite

Chair of the Supervisory Committee:
Chair Linda Shapiro
Computer Science and Engineering

Artificial intelligence and machine learning are rapidly advancing the ability of computers to see, listen, understand, and interact in the real world. Data is crucial to training these systems, and the quality of the data, the source of the data, and how it is collected and labeled are as important as the data itself for building systems that are effective, robust, and ethical. Computer vision, as a subset of machine learning dealing visual processing, is at a point where there are powerful algorithms and ample computing power, but there is a bottleneck of high quality labeled data available to train these algorithms. At the same time, there is an untapped source of data available; humans possess innate visual processing abilities, cameras are ubiquitous, and the humans using these systems on their camera-enabled mobile devices can be invited to participate as active teachers of these systems.

When computer vision moves from the research lab to the real world, there is often a mismatch between data used by researchers and the practical requirements of real-world applications. Issues of representation, reliability, and how the data was sourced bias datasets in ways that are often at odds with what is seen in the real world. Sometimes the conditions match well enough to build a useful application; other times, edge cases fail to work, or algorithms cause real harm (e.g., by perpetuating problematic human biases). There is a need to develop new ways of gathering data for computer vision that better match what is encountered in the real world and enable successful application of computer vision technology.

This thesis presents *Crowd-Driven Computer Vision*, a framework for crowdsourcing new datasets by building interactive computer vision systems that let members of the crowd test and improve these systems, and accumulate data in areas they care about. This framework has been evaluated through three projects, all deployed as games, in three different areas of computer vision, ranging from photogrammetry, to geometric reconstruction, to facial expression recognition. In each project, novice users interacted with deployed systems and collected data that was highly relevant to the system and would be unrealistic to collect in other ways. In addition to computer vision, this work draws on research in human-computer interaction for crowdsourcing complex work, and on game design as a lens for building systems that are interactive, engaging, and teach players to understand complex systems. Crowd-driven computer vision is an effort to center humans in the development of this technology.

TABLE OF CONTENTS

LIST OF FIGURES

ACKNOWLEDGMENTS

This degree took me almost twelve years to complete so I've had a long time to accumulate people and communities for whom I am incredibly grateful.

First, I'd like to thank my committee, Linda Shapiro, Steve Tanimoto, Daniel Avrahami, Alex Colburn, and Cecilia Aragon for coming together to help me make it to the end. Thank you for their support, advising, and recognition of the value of my work. Individually, thank you to Linda for being a steadfast presence in GRAIL, to Steve for running the 590D seminar (Computer-Based Learning Environments) where I had a joyous time developing the flashcard-learning game Picard, to Daniel for providing invaluable input at crucial points along this journey to propel me forward, to Alex for being an enthusiastic, knowledgeable, and supportive colleague and friend in research and other aspects of grad school, and to Cecilia for being an awe-inspiring researcher and rounding out my committee as my Graduate School Representative.

Thank you to my peers who made (many parts of) grad school immensely enjoyable. To my housemates in the MAKA House and the Lighthouse, to my colleagues in GRAIL and the Center for Game Science, and to my friends who upheld the Thirsty Thursday tradition for many years, we had many adventures and made many fond memories together! I'm worried by listing names I'm going to forget someone important or mention people in multiple places, but here goes, each of these fine humans made me glad to be at the University of Washington in CSE: Karl Koscher, Alice Neels, Jonathan Beall, Amanda Klaus, Matt Kehrt, Cynthia Matuszek, Tammy Denning, Alex Jaffe, Alex Colburn, Kevin Wampler, Paul Pham, Erik Andersen, Yun-en Liu, Rahul Banerjee, Eric Butler, Seth Cooper, Fay Shaw, Sandra Fan, Mike Chung, Qi Shan, and Ricardo Martin Brualla.

Thank you to everyone I collaborated with, especially senior grad students Seth Cooper and Noah Snavely who shared their research with me and helped chart the direction of my own research. Thank you to Erik Andersen for navigating how to make games as a method of computer science research with me in the first years of grad school and for being an avid player and fan of PhotoCity. Thank you to Rahul Banerjee for letting me share my research PointCraft with him (we wrote a wonderful and award-winning paper together!) and for being a supportive friend the whole way through. Thank you to everyone who I collaborated with on Picard, we had great synergy as a team and that project gave me so much enjoyment. Thank you to the other student collaborators on my research, Dun-Yu Hsiao, Sylvia Tashev, Nadine Tabing, and Channie Wu.

Thank you also to the support staff at UW, especially Stephen Spencer in GRAIL for taking care of servers, storage, clusters, and databases to host my games and giving me an opportunity to learn all about deploying scalable systems. Thank you to the CGS staff Barb Krug, Marianne O'Keefe, Christy Ballweber, and Kate Fisher for helping me make my games awesome and for making me feel proud to be part of CGS for the beautiful games they helped produce.

Thank you immensely to the graduate advising staff, especially to Lindsay Michimoto for helping me navigate the highs and lows of the grad school journey, and to Elise deGoede Dorough for welcoming me back into the department after my break and for guiding me through the final logistics. These women are amazing at their roles and the department is lucky to have them.

Outside of UW, there was a whole host of supportive characters from other parts of my life. Thank you to my wonderful friends from undergrad at UC Santa Cruz, especially my undergrad roommate and BFF Elizabeth Uselton for moving up to Seattle, living out her own dreams, and giving me a couch to crash on and kitties to snuggle when I was in town for my defense. Thank you also to Ben Samuel, my dear friend and programming partner

for many wonderful projects and games, who has understood and encouraged me every step of the way and has been a continual source of enthusiasm and inspiration. I'm so glad we ended up working in related fields in grad school, attended many of the same conferences, repeatedly overlapped in Santa Cruz, and have been able to share ideas for school, work, and life as we've matured from wide-eyed college freshmen to, like, grown-ups or something. From that same dorm freshman year, thank you to my dear friend Kay McKelly for being my doula and helping to deliver my non-thesis human baby, while also empathizing with all the challenges of grad school. These long-term friendships and ties back to Santa Cruz meant and continue to mean the world to me.

In addition to the people above, thank you also to the academic community at UC Santa Cruz, especially everyone that is now part of the Computational Media department. This group was like an academic home away from home that offered different perspectives on what it meant to do research in the field of game design and has proven itself to be full of creativity, excellence, ingenuity, inspiration, and respect. Thank you for little things like being welcoming when I came to visit my partner Adam, for big things like inviting me to teach the class on Interactive Narrative, which I learned so much about through being the instructor, and for personally monumental things like being my audience for my practice defense. My path through grad school would have been completely different without access to this community.

Thank you to the researchers at FXPAL for hosting me as an intern multiple times, especially Eleanor Rieffel, Don Kimber, and Daniel Avrahami. Eleanor Rieffel overheard me mention I was working on 3D reconstruction at a She's Geeky conference (thank you to Kaliya for making those events happen!) and invited me to work on projects involving augmented reality and tangible user interfaces with herself and Don Kimber. A few years later, Don invited me back to work on projects involving collaborative photography. This second internship was where I reconnected with Daniel who helped me transform some of my

unpublished work into two award-winning papers that now constitute two meaty chapters of this dissertation. FXPAL represented my ideal combination of computer vision and human-computer interaction work in a creative and supportive environment where I was able to explore the value of my own ideas.

Next up, thank you to my colleagues at Grokstyle, especially the founders Kavita Bala and Sean Bell. It was an honor to work with them on what felt like technology of the future and I felt myself learn and grow by leaps and bounds working there. Thank you to all my colleagues and especially to Beth Snavely for making the initial connection between me and the team. That opportunity was the most enjoyable of my career thus far.

Most of the time I worked for Grokstyle, I worked remotely out of the NextSpace coworking space in Santa Cruz. Thank you so much to that incredible community, to my closest friend there Jules Holdsworth, and to the strong and powerful women who run the whole show there, Maya B Delano, and Jennifer Hamilton. Thank you also to Bennett Roesch, who was the first friend I made at NextSpace and opened the flood gates to befriending everyone else. During one instance of his social connectivity games "what do you want?" I recall answering in 2016 or 2017 that what I really wanted was to be done with grad school. I got what I wanted, so now it is time to figure out what I want next!

Thank you to the intellectual communities I had access to while on a break from academic, including the Asilomar Microcomputer Workshop, Hackers, SuperHappyDevHouse, and the Invisible College. These groups shifted my perspective to a wider view of what I could and should work on and what a long and fulfilling career and life could look like.

Thank you to a bunch of random people on Twitter whose tweets and discussions sparked ideas, validated feelings, and taught me new things. Thank you to the unexpectedly large number (over 11,000) of people who liked my tweet about successfully defending—what a proud extended internet family I didn't even realize I had! Thank you especially to the subsection of twitter talking about challenges in grad school, triumphing over adversity, and

caring for one's mental health. In an effort to be transparent and reduce social stigma around mental health as I have seen done by others, thank you immensely to my therapist who for the past year has helped me unpack and sort through the emotional baggage of leaving and coming back to grad school.

Thank you to my parents Vicky and Don Tuite for their constant love, understanding, and encouragement for my entire life. What great parents! There is really no way to fit everything wonderful about them here so I'll just focus on how they provided me with many great opportunities (my dad gave me a camera, my mom got me a GeoCities account and got me into programming, and those childhood experiences evolved into me working on online collaborative photography and computer vision!) but never forced me down a specific path; for as long as I can remember, they have trusted me to navigate my own path while providing a wise and nonjudgmental respite whenever I needed one.

Finally, thank you to my husband Adam Smith and my daughter Myriad Tuite. Adam made this whole grad school thing look pretty interesting over a decade ago. I'm grateful for his companionship in this entire journey, through a long distance relationship, countless projects together, adopting our kitties JPEG and MPEG, moving in together, getting married, moving around a bunch, figuring out our careers, and raising our wonderful daughter Myriad. Thank you especially to Myriad for being a curious, empathetic, and silly kid who fills my heart to the brim.

DEDICATION

To my family, especially my daughter Myriad and my husband Adam

Chapter 1

INTRODUCTION

1.1 Motivation

Humans possess extraordinary abilities to process and understand visual information, and over the last several decades, human-kind has been working to impart this knowledge and expertise to computers. Through computer vision research, progress is being made to emulate human visual perception, e.g., a robot 'seeing' through cameras and interacting with the physical world. Computer vision applications are also being developed to augment human abilities and allow people to access information captured through cameras, e.g., using a mobile application to identify a species of flower, translate text to a different language in real time, or fetch purchasing information about a piece of furniture.

Despite the progress, there is still a long way to go before computer vision reaches its full potential. There are very few circumstances in which end users are allowed to directly influence computer vision, which is a missed opportunity considering the abundance of cameras and processing power (on device and in the cloud) that many users have access to and their unique vantage points on the world around them. Augmented reality is poised to take advantage of those cameras and computing power, but is currently missing the visual understanding element of what is happening in the real world in order to augment reality in a more meaningful way.

I believe that involving end users, those who are pointing the cameras at the world, in the process of teaching visual understanding to computers is key to developing applications that represent the full potential of computer vision. Data is the driving force behind much of computer vision, as well as a hindrance to progress. This is where my thesis primarily seeks to involve humans: in the dataset collection and curation process.

Applying computer vision in the real world, after its incubation in a research lab, often reveals limitations, unforeseen consequences, or biases of the training data. With photography especially, the images can be captured in different ways (e.g., different types of cameras, different lighting conditions, different viewing angles) or for different purposes (e.g., personal, professional, sentimental, informational) than what fits the real usage of the application. A dataset may represent information in an unbalanced way (e.g., many photos of a common bird species and few of a rare species) that causes an algorithm to perform poorly on the underrepresented (e.g., a facial recognition system that misclassifies black women with a higher error rate than any other group [18]). Sometimes the conditions match well enough to build a useful application, but other times there can be many boundary cases that fail to work, or even situations where algorithms cause real harm (e.g., by perpetuating problematic human biases).

There are several major paradigms for collecting and labeling data that contribute to the problems listed above. The first two paradigms I discuss are about collecting data, and the third is about labeling data, and they can be used in combination.

The first paradigm is that of *found data*, of using data scraped from public online sources. Immediately, this raises the issue of differences in how and why these images were captured (e.g., vacation photos vs. photos taken explicitly to build a 3D model). While this data might be enough to bootstrap an application, it will have gaps in coverage that impair how well the application works. Because the original contributors of the data had a different purpose in mind (e.g., social sharing of photos vs. training a computer vision algorithm), those gaps may never be filled. Additionally, the contributors of this data are likely unaware of what their data is being used for and are unable to provide explicit consent.

Another paradigm is the *centralized accumulation of private data*, in which user data from a particular platform such as a social media site is aggregated and used en masse to train powerful algorithms. This is exemplified by Facebook's highly accurate results in facial recognition [188] due to the amount of semi-labeled face data at their disposal. This paradigm has a similar problem to the previous found-data paradigm of using data captured with a

different intent than training computer vision applications, but the scale of the solutions and the types of applications (e.g., ones that further support the platform) can offset this. On the other hand, this paradigm shapes the kinds of problems that the corporations behind these platforms choose to solve, preferring technology that benefits the largest number of people in the most general way. This concentrates the power in the hands of the select few performing research at these institutions and misses opportunities for narrow, deep, high-impact applications, because they might benefit fewer users.

The third paradigm, the *Human Processing Unit model (HPU model)*, has to do with how this data, both public and private, is annotated via microtasks. This *HPU model* uses interchangeable workers to perform high volumes of short, low-skill tasks, which often consist of quick judgments about images that are easy for humans but hard for computers. The idea is that the human is a stand-in for a computational task that, once trained on human-provided examples, a computer will eventually be able to do automatically. This labeling strategy, while scalable, misses opportunities to share additional context about a problem that would help workers perform better, or make use of their individual expertise. It does not provide an environment in which users can directly develop their skills, or allow them to use their own data to fill in gaps that they can identify for themselves.

The number of people involved in designing computer vision technology, and even deciding what applications to build, is relatively small compared to the number of people whose data and labor provide the foundation of this technology. What if these people, the intended users of these applications, with their eyes, human experience, and cameras, could play a more active role in defining the future of computer vision?

In the natural sciences, only a small number of people are typically involved in scientific inquiry and advancing science. However, the last decade gave rise to scientific discovery games that engage ordinary people in the scientific process as citizen scientists. What if we had citizen *computer* scientists? Of all areas of computer science, the general population, with its eyes, brains, cameras, and curiosity, is well situated to take on computer vision specifically, and to collectively teach machines this skill in a more intentional and active way.

My dissertation presents *crowd-driven computer vision* as an alternative paradigm to those listed above, one that puts the crowd in the driver's seat. Crowd-driven computer vision is a way of involving the crowd in the process of teaching computers through simultaneous data collection and labeling, as well as ongoing evaluation and improvement of computer vision systems in real world contexts.

In order to use computer vision to solve more problems and hasten its successful application to new areas where it is greatly needed, it is important to broaden the pool of people who can gather data, decide what kinds of data to gather, and even to determine what kinds of problems are important and worthy of solving. Many problems are worth solving, but they might not be "worth it" at the right scale of money or impact for the people and organizations that have the resources and expertise to work on them.

Ideally, many people would be able to use computer vision for their own applications in the same way that programming and web development are increasingly accessible. Tools, frameworks, and learning resources exist to help individuals build computer vision applications, including Javascript libraries that run face detection and object recognition algorithms in the browser and mobile development frameworks like CoreML from Apple that make it easier to build and deploy mobile applications that use custom machine learning models bootstrapped by pretrained shared models.

However, even with these resources available, computer vision datasets and systems are developed in isolation. As with the world wide web, the presence of other websites increases the value of individual websites and the shared ecosystem is more valuable as a whole than the individual parts. Computer vision can also benefit from a varied, connected, and shared ecosystem, especially of data, that many people work together to build and that strengthens the utility and value of individual datasets curated for specific purposes. There is a need for tools, infrastructure, systems, and design patterns for these systems that support cooperative work on collecting, cleaning and curating shared datasets in computer vision and in designing and evolving the algorithms and applied systems that use that data.

1.2 Contributions

I have developed the Crowd-Driven Computer Vision (CDCV) framework for building CDCV systems, which are interactive computer vision systems that make deep and engaging use of crowdsourcing. The framework consists of four properties required for CDCV systems, including 1) the integration of a core data-driven computer vision technology, 2) a data space for that technology that can be represented as a map (even an abstract map), 3) interaction that enables a crowd to interactively and collaboratively map this space, and 4) feedback from the system that informs crowd participants about the quality of their contributions. I claim that the properties of this framework, when applied to the design of a crowdsourcing system for computer vision, can lead to certain outcomes including 1) accumulating data in areas of user interest, 2) the possibility of collecting new types of datasets, 3) engaging the creativity, skill, and diversity of the crowd, and 4) motivating and focusing crowd effort towards a shared purpose. These outcomes can, in turn, contribute to the development of computer vision applications that are more useful, robust, effective, and fair by sourcing data for these applications.

I demonstrate the CDCV framework through a set of experimental systems that I have designed, built, and deployed. The three systems described in this work are PhotoCity, a game for reconstructing 3D models from photographs, PointCraft, a tool for interactively tracing point clouds to build clean 3D geometry, and The Meme Quiz, a game for testing and teaching a facial expression recognition system on diverse facial expressions. Each system possesses the four CDCV properties and exhibits the outcomes of the framework. The deployment of these systems has enabled new types of datasets to be collected, restructured batch algorithms to be incremental and collaboratively controlled, and engaged crowds of people in the data collection and training process by inviting them to target CV systems toward things they care about personally.

The CDCV framework draws inspiration from three fields: human computer interaction (and crowdsourcing research in particular), game design (games with a purpose, in partic-

ular), and computer vision itself. This work makes contributions in each of these fields. In HCI, I propose new methods of crowdsourcing that enable complex work by training people as they participate. In game design, I propose new design patterns for games with a purpose, in particular the idea of building game mechanics out of computer vision and machine learning. In computer vision, my work has collected new datasets that could not have been created in any other way but through interactive crowdsourcing systems that the crowd has the power to direct.

1.3 Outline

The structure of the remainder of this document is as follows: In Chapter 2, I discuss related work in computer vision, crowdsourcing, collaborative creativity, and games with a purpose. Chapter 3 describes the Crowd-Driven Computer Vision framework itself, including an overview of how each of my three systems (PhotoCity, PointCraft, and The Meme Quiz) exemplifies the framework. The following chapters explore each system in detail. Chapter 4 examines how feedback in PhotoCity taught players to understand the underlying computer vision system and develop strategies to contribute more effectively, while collecting a large dataset of highly relevant imagery. Chapter 5 discusses how, in PointCraft, the use of a familiar game user interface for navigation and interaction allows players to directly trace, fill gaps, and construct clean geometry out of the noisy, partial point clouds produced by PhotoCity. Chapter 6 moves away from 3D reconstruction to the domain of facial expression recognition with the game The Meme Quiz, and discusses how users actively improve the underlying facial recognition system over time while collecting facial data that represents a diversity of expressions and realistic capture conditions. The final chapter, Chapter 7, concludes this dissertation with a discussion of contributions and implications for future work.

Chapter 2

RELATED WORK

This chapter summarizes related work in computer vision systems that use *found data* from public online sources, as well as general crowdsourcing techniques, collaborative creativity, and games with a purpose. Section 2.1 outlines computer vision systems that have been built to take advantage of the abundance of visual information available online, but these systems and datasets are limited by what data can be *found* versus what data needs to be collected, curated, and even created from scratch. Each system described in this thesis (PhotoCity, PointCraft, and The Meme Quiz) is built around a computer vision system for a different sub-domain of computer vision and is designed to collect data that fills gaps in the default dataset for that system. Section 2.2 describes how crowdsourcing has been used to label large-scale datasets for computer vision and covers additional efforts in crowdsourcing to enable deeper and more complex work. Creativity is an element of deep, complex work, and Section 2.3 looks at online collaborative creativity systems, in and outside of crowdsourcing purposes, built for casual users. Games with a purpose, covered in Section 2.4, represent the original efforts to label data for computer vision at scale, and have continued to evolve over the past 15 years. Each of my systems is also a deployed game with a purpose, but is designed to foster more complex, collaborative, and skilled work than many traditional games with a purpose. Over all, my work synthesizes ideas from crowdsourcing complex work, and collaborative creativity, and designing games with a purpose.

2.1 Found Data and Computer Vision

The availability of online imagery has made a significant impact on computer vision research. The technology used in PhotoCity (Chapter 4) to build 3D point cloud reconstructions from

photos was an adapted version of the Structure from Motion pipeline from Snavely's Photo Tourism [181] project. Photo Tourism was one of the first research projects to make use of large quantities of publicly available online image data of places to build 3D representations of those locations, including the estimated position of every camera. It showed how images from many different sources could be analyzed together, despite the myriad photographers, cameras, and capture conditions involved. In the same vein, similar collections of photos have been used to understand how people move through places [180] and how places change over time [127].

The availability of online imagery at scale has also influenced object detection and recognition. ImageNet [44], MS-COCO [121], and OpenImages [113] are just a few of the datasets collected from photo sharing sites and internet search results. Datasets have also been collected around faces, including PubFig [112], and Labeled Faces in the Wild [81].

Researchers have designed innovative systems to process this real world data, from reconstructing and animating 3D models of faces from just photos [185], to linking text descriptions on Wikipedia to navigable 3D spaces [160], to understanding design through style embeddings in furniture [11] and fashion [129, 203]. These systems represent algorithms that are robust to the variability inherent in imagery taken by many different people and devices around the world.

That robustness makes these systems very powerful, but when these research systems are taken out of the research lab and into a real world setting (such as applying the Photo Tourism software to PhotoCity), it is often revealed that the datasets have gaps and biases and the algorithms are not designed to be updated easily when these gaps and biases are discovered.

Facial recognition is one area where dataset biases, and the ethics surrounding the collection of these datasets from online sources, are particularly notable. Keyes discusses how technology for automatic gender recognition, including the data, labels, and algorithmic inputs and outputs, operationalizes gender in a trans-exclusive way, treating gender as binary and immutable, which essentially erases transgender people from the design concerns of this

technology in a way that is discriminatory and potentially harmful [96]. The project Gender-Shades [67] exposes how commercial facial recognition systems that classify gender perform significantly worse on faces of dark-skinned women than other gender and skin type combinations. To proactively call attention to dataset biases and prevent potentially harmful uses, Gebru et. al. propose Datasheets for Datasets [18], a practice of including documentation about the creation, composition, characteristics, and intended uses in the release of every dataset.

Algorithms designed to process real world images from the internet have the advantage of working in a wide variety of conditions, from different lighting, poses, camera quality, photography style, and photography purpose. What these algorithms tend to lack, however, are avenues for many different people to interact with them in a variety of ways. To fill in the gaps of found data, we need to redesign computer vision algorithms to be more interactive and to be collaboratively operated and improved.

2.2 Crowdsourcing

Large-scale crowdsourcing through distributed, online platforms has been a key part of cleaning and labeling online data for computer vision purposes. Games with a purpose (covered in Section 2.4) represent the earliest approach to crowdsourced labeling at scale, but games are time consuming to design and produce. A more common approach that can accomplish the work more directly than through a game is paid crowdsourcing. Within computer vision, crowdsourcing through paid microtask platforms such as Amazon Mechanical Turk (AMT) has been used to build, among other databases, the large-scale hierarchical image database ImageNet [44], a database of abstract images with ground truth semantic information [226], and OpenSurfaces [12], a database of annotated surfaces and materials in real world photos. AMT has even been used as a recruiting platform for a game for identifying what part of a bird to use to distinguish its species [45].

As with games with a purpose, there is a spectrum of crowdsourcing that spans simple tasks that anyone can easily complete in a matter of seconds, to complex tasks that require

deeper understanding of the context, greater skill, and more complex interactions, but that can also have more profound impact. Within the HCI community, researchers are experimenting with the design of many different crowdsourcing pipelines, many of which are built on top of AMT. Soylent [14] is a system that allows the crowd to assist with the complex and creative task of writing. Cascade [27] allows crowds to create taxonomies without expert knowledge or a global understanding of the objects to be categorized. Mobi [224] is an example system that enables complex tasks with global constraints. Designing successful crowd pipelines is a time consuming process, and TurKit [122] is a programming toolkit that enables the rapid exploration and prototyping of such pipelines. AMT has even been used for near-real-time responses as with VizWiz [16], a mobile application that lets a blind person take a photo, ask a question about the photo (e.g. "what does this can of food contain?" or "what denomination is this bill?"), and quickly get back a spoken answer.

Outside of AMT, crowdsourcing researchers have built platforms that explore new ways for members of the crowd to collaborate in complex and meaningful ways. Such systems include ConsiderIt [107] for public deliberation, TweakTheTweet [183] for augmenting how microblogging platform Twitter is used for crisis reporting, and Dynamo [165] for enabling collective action of crowd workers on issues of direct relevance and importance to themselves. Project Sidewalk [164] is a platform and interactive tool for virtually inspecting and labeling accessibility information about the physical world through Google Street View imagery. Like these systems, the games presented in this thesis constitute their own crowdsourcing platforms that exist outside of the microtask platform of AMT.

A common approach in crowdsourcing, especially on AMT, is to think of workers as *human processing units* [41] (HPUs) and to design crowd work flows inspired by computational work flows in distributed computing. Seen this way, crowd workers form a specialized computation device with complementary strengths to CPUs and GPUs. Implicitly, HPUs are presumed to share properties with other computational devices such as working at a steady rate, repeatedly producing equivalent outputs for identical inputs, and requiring only a small amount of software (instructions) to perform a self-contained task.

Even though workers operating as HPUs are often working on data to train fully automated artificial intelligence systems, these workers rarely have AI-powered tools at their disposal. Photo Sleuth [136], a tool for identifying historical photos of civil war soldiers, is an example of a crowdsourcing system outside of AMT that uses a human-led, AI-supported approach in which face recognition is used to narrow the search space and find similar faces for a human to closely examine. This approach to crowd-AI that has humans making the final decisions is in line with crowd-driven computer vision.

In their report *The Future of Crowd Work* [101], a group of crowdsourcing experts shared concern that crowdsourcing itself will be seen as exploitative and with limited applications unless it can reliably be used for complex, creative, and collaborative work. This thesis shows how games can be used for complex and creative tasks by giving players more responsibility to understand and navigate a complex system, instead of carefully isolating them with their own individual units of work. The main advantage of exposing more of the underlying system to players is that it allows for creativity in the form of problem solving, novel contributions, and emergent behavior. The next section goes into more depth on creativity.

2.3 Casual and Collaborative Creativity

Creativity thrives in environments where people are intrinsically motivated by the interest, enjoyment, satisfaction, or challenge of the work, but common crowdsourcing platforms do not typically provide such environments. Dontcheva et. al. break down specifically why the low-autonomy, low-skill, isolated working conditions of AMT are ill-equipped to foster creativity and recommends alternative ecosystems that foster community, encouragement, and the time and space to be creative [49].

Crowdsourcing platforms designed with creativity in mind, especially collaborative creativity that could not be done by a single isolated individual, have had a large influence on my work in this dissertation and beyond. In music and visual art, crowdsourcing has been used to combine the unique talents of hundreds or thousands of individuals, distributed around the world and united online, into composite artifacts. Eric Whitacre's Virtual Choir [216] is

a project in which 185 people sang individual parts of a choral piece, and the tracks of voices were combined into a final composition. The Johnny Cash Project [134] was a website where people could hand-illustrate frames from a Johnny Cash music video using simple grayscale painting tools and play back the video with each frame contributed by a different person with a different aesthetic style.

Koblin, a creative director of the Johnny Cash Project, has released several similar crowd-sourced art experiments involving drawing vehicles (The Single Lane Superhighway), drawing sheep (The Sheep Market), drawing a small piece of a $100 bill (10,000 Cents), and singing a minuscule clip of a song (A Bicycle Built for Two Thousand). Each of these individual contributions was then combined with the others in a visual or audible mosaic, with a browsing option allowing a viewer to zoom in on the work of an individual. The value of these collaborative works of art comes from the distinct style that each individual brings to the piece and how no single artist would have been capable of such variety.

Creative projects undertaken by large groups of people online are usually designed for the participation of novice or casual users. Simple paintbrush tools on a canvas that anyone can access through a web browser or a limited color palette that makes most output aesthetically pleasing mean that anyone can jump in and start creating without pressure. Removing choices and making it harder to fail are some of the patterns explored and defined by Compton in her dissertation on Casual Creators [30]. Casual Creators are interactive systems that enable creativity for creativity's sake by removing barriers to creativity and encourage a state of creative flow.

One such barrier to creativity is the blank canvas, the moment when there are the most potential possibilities, inspiration has yet to strike, and it is hard to take the first step. An online collaborative setting can remedy this by allowing users to see and draw inspiration from others' contributions, and to even start directly from another user's work through remixing. Remixing is a key component of the programming ecosystem Scratch [156], the mass collaboration environment for young people to create and share their own interactive media. Scratch was designed with collaboration in mind, and is a powerful process for

learning [158].

Scratch, with its user base of young people, has also been evaluated alongside collaboration support tools for adults in online scientific communities. A key finding is that systems that support social creativity must facilitate sharing and play as ways to foster imagination and provide encouragement for the people involved and their budding ideas [5].

In my own work outside of this dissertation, I have developed several online collaborative creativity platforms to further explore ideas of remixing, sharing, collaborative learning, and play. Sketch-a-bit is a mobile drawing application in which a user starts drawing from a random user's drawing instead of a blank canvas, and then uploads the new drawing to the pool of images for another user to work with. Analyzing over 50,000 sketches created in this application, I identified emergent remixing behaviors, in particular that of 'deep chains' where the ideas of one individual feed into and directly inspire another individual for many levels [196].

The flashcard learning game Picard is another game I was involved in building that allowed players to study flashcard topics (such as countries and capitals) through spaced repetition, but also by drawing and sharing mnemonic pictures to help themselves and others remember certain topics. This creates a collaborative forum for learning where the creativity, cleverness, or absurdity of another player may help cement certain topics in players' minds [195].

The game systems presented in this dissertation represent explorations into collaborative creativity applied to computer vision, by allowing easy sharing work to inspire, educate, and bootstrap the work of others, by lowering the barrier to entry such that novice users can make meaningful contributions right away, and by supporting playful exploration and experimentation. In computer vision, there is more room for creativity and playfulness than has yet been explored, and my work seeks to see where that openness leads and where the creativity of the crowd can drive progress in computer vision.

2.4 Games with a Purpose

Games with a purpose (GWAPs) are a type of game in which the player's actions in the game contribute to a real-world purpose outside of the game, whether it be predicting protein structures or providing labels for images.

Luis von Ahn and Laura Dabbish kicked off the genre of "games with a purpose" [208] in 2003 with the first GWAP, the ESP Game [207]. The ESP Game paired two anonymous players online, asked them to provide labels for an image they both saw, and rewarded both players for agreeing on labels. The purpose of the game was to provide labels for images, labels such as type of scene or what objects are present, exactly the kinds of common and recognizable attributes that a player would expect their anonymous partner to also recognize. In the same vein, von Ahn and colleagues produced Peek-a-Boom [210], Verbosity [209], and TagATune [114], games with the purpose of producing tags or annotations for images or songs through creative combination of humans coming up with and verifying each other's tags. These GWAPs inspired additional games including KissKissBan [78], a game for image annotation, Happy Moths [151], a game by Citizen Sort to categorize moths by their shape and color, and OntoGalaxy [106], a 2D space shooter game for populating an ontology.

These games and game design guidelines [208] represent a narrow interpretation of purpose, where the main goal is to accomplish human computation work that may be boring and tedious by disguising it as entertainment. I explore this problematic framing more thoroughly in [192] and propose how GWAPs can be more impactful by better aligning purpose with mechanics, and by giving players something to learn. Von Ahn's more recent work includes the language learning game Duolingo [206] for training people to translate content on the web into other languages.

In an effort to aid development of games with a purpose, Siu et. al. have worked to formalize design knowledge about these games by studying the effectiveness of different reward systems [176] and different game mechanics based on collaboration or competition [177]. Siu et. al. also proposed a framework for describing human computation game (HCG) mechan-

ics [178] based on *action*, *verification*, and *feedback*, which closely map to three of the four CDCV system properties of *interactivity*, having a *core computer vision system* that can be used for validation, and *feedback*, each discussed in Chapter 3.

Citizen science games represent a way to involve ordinary people in science through game mechanics and user interfaces that are approachable and accessible by people without scientific backgrounds. In Foldit [35], players manipulate the 3D structure of a protein and try to find the most physically plausible (lowest energy) way to pack the protein together. EyeWire [100] is a game for mapping the 3D structure of neurons in the brain. The game EteRNA [115] is a puzzle game for designing RNA structures, which are then manufactured and tested. The experimental results from the lab are returned to the players to improve their designs. The players of each of these games have the potential to be personally invested in the purpose, in learning about the underlying science of each game, and developing skill and intuition that allows them to make meaningful contributions and become scientific collaborators. Galaxy Zoo [153] and Zooniverse [173] have taken citizen science a step further by creating an online platform for many scientists to publish tasks with which they need help, and crowdsource hundreds of thousands of people into the scientific process.

While not explicitly a game, the web-based annotation tool LabelMe [161] provided a way for the public to easily label images at the level of object class and object segmentation. Mobile applications such as LeafSnap [111], iNaturalist [201], and eBird [217] have provided additional ways for users to take photos that contribute to (and test out) computer vision techniques, at least in the animal and plant species identification realm.

This involvement of the public in the forefront of science through games is what has inspired my work on games in computer vision. With the proliferation of cameras, human visual expertise, and computer vision technology reaching more peoples lives, these citizen science games can and should be extended into the science of teaching computers to see.

2.5 Conclusion

This chapter summarized related work in computer vision systems designed to work with in-the-wild found data from the internet, crowdsourcing, collaborative creativity, and games with a purpose. Although games with a purpose have historically been used to crowdsource labels for computer vision, my work extends the reach of games with a purpose by integrating ideas from crowdsourcing complex work and from systems designed to support collaborative creativity. This allows for more complex games that can collect new datasets to fill in the gaps of found data while teach players about the underlying computer vision systems and allowing them to drive those algorithms in directions of applications they care about. The next chapter goes into detail about these ideas as it describes the *crowd-driven computer vision* framework.

Chapter 3

CROWD-DRIVEN COMPUTER VISION

This chapter introduces the crowd-driven computer vision (CDCV) framework, a conceptual framework for designing interactive computer vision systems that empower crowds of people to collaboratively collect and label data, evaluate, and improve a given computer vision application.

Under this framework, there are four properties that must be present in a system for it to be a crowd-driven computer vision system: 1) having a core computer vision technology to provide focus for that system, 2) having a defined and mappable data space, 3) having interactivity that allows the crowd to collaboratively map the data space, and 4) providing feedback from the system to users explaining the effects of their contributions.

I claim that these properties of crowd-driven computer vision systems support the following CDCV outcomes including 1) accumulating data in areas of user interest, 2) collecting new types of datasets, 3) engaging the creativity, skill, and diversity of the crowd, and 4) motivating and focusing crowd effort towards a shared purpose. It may be possible to build a CDCV system with the required properties that fails to meet these outcomes, in which case perhaps the design of one of the properties or how they are integrated together needs to be improved. Such outcomes represent a path towards developing computer vision that is fair, transparent, and ethical, where ordinary people play a role in the design and instruction of computer vision technology while also learning about how it works, and where the application of computer vision technology can solve end users' problems by collecting data that aligns with users' interests.

My three systems described in later chapters, PhotoCity, PointCraft, and The Meme Quiz, are representative of these properties and outcomes, each in its own way.

3.1 The CDCV Framework

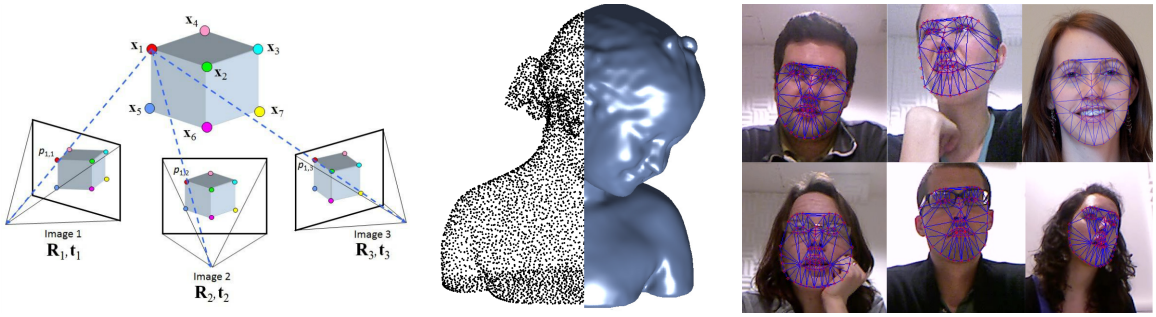
A *crowd-driven computer vision* system is an interactive crowdsourcing system that engages the crowd in collaboratively collecting data and improving a computer vision system. The CDCV framework defines four properties of CDCV systems: 1) having a core computer vision technology to provide focus for that system, 2) having a defined and mappable data space, 3) having interactivity that allows the crowd to collaboratively map the data space, and 4) providing feedback from the system to users explaining the effects of their contributions. I now explain each property in detail.

3.1.1 A Core CV System that Needs Collected Data

At the core of every CDCV system, there is a computer vision technology or application that is driven by data. More data means a bigger result can be assembled (e.g., reconstructing 3D geometry from thousands or millions of photos), better algorithms can be developed (e.g., understanding new edge cases in point cloud reconstruction), and better models can be trained (e.g., more accurate image classification systems). Figure 3.1 shows three examples, each pertaining to the computer vision technology behind the three projects in this thesis. A CDCV system is built around one computer vision technology in order to collect data for a specific application of that technology, or for many applications that share that technology.

The purpose of a CDCV system is to collect new data for that core technology by exposing an interactive slice of that technology to users and focusing especially on the data: what data has been collected so far, what data still needs to be collected, and how adding additional data grows the result or improves the system.

A computer vision system that is deployed and used by the crowd, but does not accumulate data is not a CDCV system because it lacks additional properties of CDCV systems regarding user agency to find and fix issues and the system evolving through accumulating new data. Examples of systems that can be used without accumulating data include the camera-based translation from Google Translate (based on Word Lens), Microsoft's HowOl-



(a) Structure from Motion (b) Point Cloud Reconstruction (c) Facial Detection and Recognition

Figure 3.1: Here are three examples of data-driven computer vision systems that correspond to the three CDCV projects in this thesis: structure from motion (PhotoCity), point cloud reconstruction (PointCraft), and facial understanding (The Meme Quiz). (The final image, Figure 3.1(c) shows face detection results from [8], which can be used as a preprocessing step for facial expression recognition.) Each of these technologies can be improved upon with larger quantity and better quality of data, and thus can form the basis of a CDCV system or benefit from data collected through a related CDCV system.

dRobot demonstration, or Google's Art and Culture Portrait Doppelganger search. Even if usage data, including images, is captured and stored to potentially be processed at a later date, the system is not a CDCV system unless the ongoing integration of new images is purposefully designed into the system.

Identifying the core computer vision system provides focus during the process of designing the system. Rather than attempting to collect all possible image representations of the world, the narrower scope of a particular application informs the implementation of the remaining CDCV properties involving understanding and mapping the space of data, user agency, user choice, and how feedback is conveyed to users.

3.1.2 Defined and Mappable Data Space

The second property is that the data space is defined and mappable. Overall, the full data space is the space of all possible appearances of the real world and how that can be

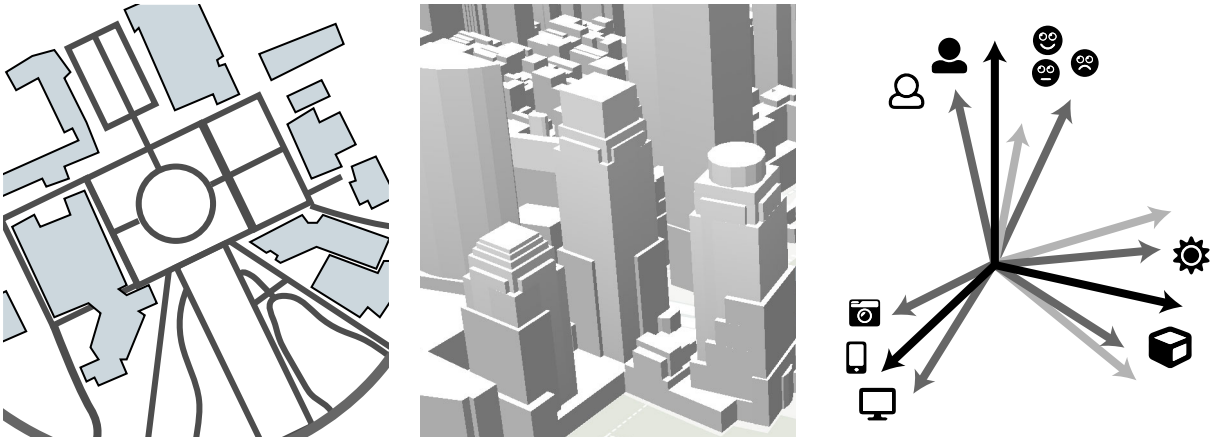
captured in different types of imagery or video. Different algorithms and applications work with imagery of specific subjects, e.g., buildings, faces, people, interior scenes, street level imagery of roads, aerial imagery, and more. Within these groups, there are dimensions related to how the imagery is captured including what kind of cameras, sensors, lenses, as well as lighting, pose, occlusions, and image quality. There are additional dimensions related to a specific subject, such as weather, season, traffic, or construction for photos of building exteriors, and age, ethnicity, hair style, expression, and accessories for faces. It is not that every dimension needs to be tracked and labeled, but there needs to be some thought put into which dimensions are relevant and how specific dimensions are constrained. Dimensions can even be machine-defined, as in automatically extracted histogram-of-gradients (HOG) features used in The Meme Quiz [194] for comparing faces, or the high dimensional space of a deep learning style embedding [11].

Defining this space allows the space to be mapped. It can be a literal map, e.g., a geographic map of the GPS coordinates of images (Figure 3.2 shows the locations of PhotoCity images) or a map that is labeled across certain dimensions (e.g., age vs. expression vs. ethnicity for faces). The map can also be abstract, perhaps a high dimensional space that may or may not have a meaningful visual representation. Figure 3.3 shows three representative maps for each project in this thesis, while Figure 3.4 shows different examples of abstract maps in other computer vision applications.

Mapping data along defined dimensions helps to reason about data coverage, but it is also possible to think of mapping and coverage in terms of algorithmic behavior and evaluation. For example, a face dataset may have light skinned faces across all age ranges, but have sparser coverage of certain age ranges among dark skinned faces, and this may affect the performance of algorithms trained on this data on certain types of input data. Even if the parameters of a training dataset are not known because the dataset itself is private, the behavior of an algorithm trained on that data can be mapped, as in the GenderShades evaluation of commercial facial recognition software on different types of faces, which revealed the lowest performance of accurate gender recognition on dark skinned female faces [18].

Figure 3.2: Some data spaces may be mappable through literal geographic maps, as with PhotoCity. The goal of PhotoCity was to collect dense, ground-level photographic coverage of urban areas that could be used for 3D reconstruction, gaming, cultural preservation, navigation, fostering further research, or a variety of other purposes. This figure shows the final computed positions of many of the photos collected through PhotoCity overlaid on a map of the University of Washington campus. (The latitude and longitudes of each photograph were computed automatically through the game and positions that seem unnatural may be due to registration errors or the alignment of the oblique angle aerial imagery.)

Mapping algorithmic behavior can take the form of understanding what areas of the data space are known to work well, and what areas are known to work poorly. For example, new photos that are similar to existing photos in a dataset are likely to work well in any number of applications, e.g., registering a photo to a 3D model via similar photos, or recognizing a face by comparing it to existing photos of the same person. Likewise, photos of textureless interior walls and reflective building surfaces do not work well for Structure from Motion, which relies on textures to identify and match feature points in photos. Knowing what works and what does not also gives rise to what *might* work, and can give insight into whether simply collecting more data with better coverage can make those boundary conditions work,



(a) Geographic Map (b) Possibility Space of 3D Geometries (c) High Dimensional Space

Figure 3.3: The map of a CDCV data space can take many forms, from a literal geographic map, to a representation that is more conceptual or abstract. Figure 3.3(a) depicts a geographic map, e.g., the type used in PhotoCity. Figure 3.3(b) depicts the space of possible 3D geometric reconstructions, e.g., the modeling space in PointCraft. Figure 3.3(c) depicts an abstract map of a high dimensional space of properties of capture combinations for a domain like facial expression recognition: lighting, pose, type of capture device, properties of a subject's identity, and properties of a subject's expression or temporary appearance. The Meme Quiz focused on collecting data in under-represented regions of this space.

or if algorithmic changes are necessary. Figure 3.5 illustrates this concept. The key is being able to identify the boundaries of known space and work to extend those boundaries. Of the three systems in this dissertation, PhotoCity is the project that most directly explores this concept, by automatically identifying the edges of 3D reconstructions and directing users to take photos at these edges in order to grow the most new geometry that still connects to existing geometry.

The manifestation of how a data space is defined and mapped will differ for different data domains and applications. Each of my example CDCV systems has a unique data representation for its own domain, but what they share is how exposing this map of the data space in some way allows reasoning about this map to take place, and for interactivity that explores and fills in areas of missing coverage.

(a) Olivetti Faces (b) ImageNet Embedding (c) Activation Atlas

Figure 3.4: This figure shows abstract map representations of data spaces of other research projects, which helps to understand where these data spaces are densely populated and well understood, versus where they have gaps. Figure 3.4(a) shows a T-SNE embedding of the Olivetti Face Dataset [125]. Figure 3.4(b) shows a 2D t-SNE visualization of 4096-dimensional embedding CNN of sampled images from INLSVRC 2012 [88]. The computed arrangement of this map clusters similar images together, e.g., yellow shapes on the bottom left and furry mammals on the bottom right. Figure 3.4(c) shows an activation atlas, a visualization of the activations of an image classification network [21].

3.1.3 User Interaction for Empowered Data Collection

The third property of a CDCV system is that it is interactive in a way that relates to understanding and mapping the data space as that data space is defined. The type of interaction should fit between the work of skilled professions with highly specialized tools and simplistic micro work that utilizes very little skill, contextual knowledge, or creativity. This spectrum is illustrated in Figure 3.6. This interaction will often take the form of users contributing new data that is relevant and valuable to the underlying computer vision system, but it can also mean identifying and fixing flaws in the map itself, and contributing metadata.

Importantly, the interaction provides insight into the computer vision system and data space map so that when a user contributes new data, that user does so with that context in mind. This is different from accumulating data as a side effect of an unrelated interaction,

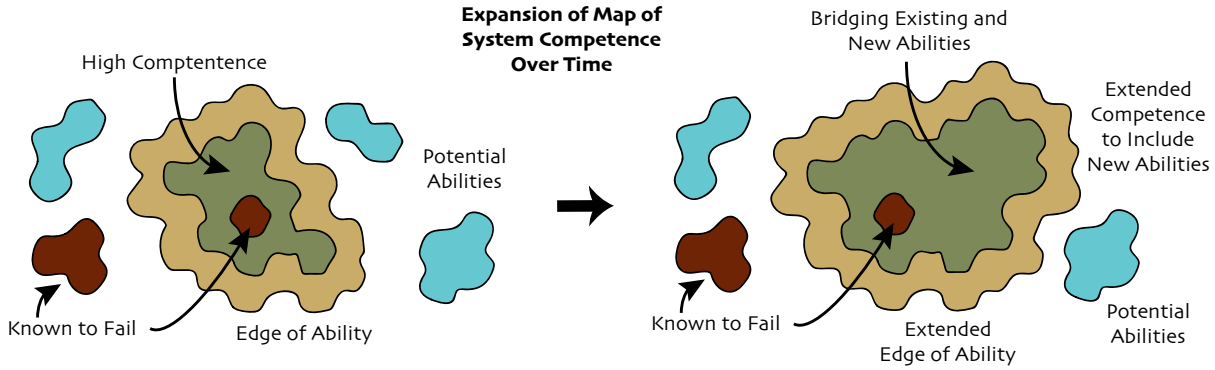


Figure 3.5: In addition to the data space itself, a map metaphor can also be used to understand the competence of a computer vision system, and how to develop that competence over time. This figure shows a map of system competence at two points in time. Where the data space has good coverage and the system performs well, there will be areas of high competence. There will also be areas of low competence, e.g., where properties of the data space in that region make it algorithmically difficult. There will also be a region that is on the boundary of what the system is capable of, where the system might perform with lower confidence or make occasional mistakes, e.g., when processing new data that is related to what is already known, but also contains new information. Incorporating that new data into the system can extend the system's capabilities and expose new boundaries of what is possible. This expansion can bridge gaps between existing and new abilities to reach potential abilities outside the original scope of the system.

e.g., posting photos on social media with the intent of sharing information with friends, while behind the scenes, that data is aggregated and used to make progress in artificial general intelligence. Through their interactions, users should be able to understand what computer vision system their data is being used in and how they are actively mapping the data space.

As mentioned in the introduction, a reason for focusing this framework on computer vision is that the prevalence of photography and cameras, especially on internet-connected mobile devices, allow users to contribute information-rich images and videos at the click of a shutter button with low effort. However, using the metaphor of mapping a space, there are many other ways to contribute to a mapping effort besides charting new territory; adding additional information, identifying and correcting errors, or updating a map to reflect changes, are all

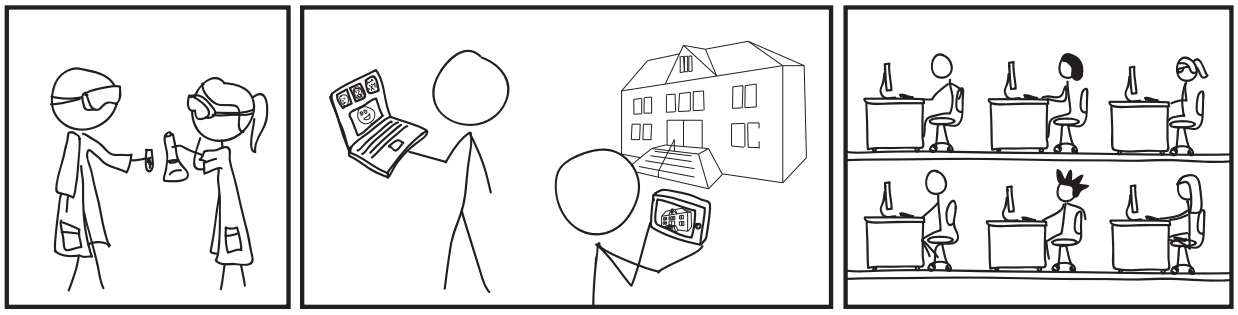


Figure 3.6: The interaction afforded by a CDCV system is what allows a crowd to collaboratively map the space through contributing new data and correcting discrepancies. The type of interaction, e.g., taking photos and receiving and providing feedback (center), should fit between skilled professions (left) and simplistic micro work that uses people as *human-processing units* (right). CDCV is meant to broaden the pool of people who can make meaningful contributions to an important endeavor, and to facilitate complex, collaborative, and creative and skilled work.

valuable ways to contribute that have parallels in CDCV. For example, Figure 3.7 illustrates how a user's contribution, especially if it relates to visual similarity as in The Meme Quiz, can include not just the data point (e.g., a new photo) but information about where the user thinks the data point fits into the existing mapped space. In The Meme Quiz, this takes the form of the user validating or correcting the system's guess.

Designing an interactive CDCV system around a computer vision system and its related data space allows for more transparency into the overall purpose and progress of the system. Users interacting with this system have access to this context, and they may even develop a deeper understanding or intuition of the underlying systems through repeated interaction. This is an advantage that CDCV systems have over the HPU model, where paid microtask workers have very little insight into the bigger context, including information that might improve the quality or efficiency of their work output. Additionally, another opportunity that CDCV users have that microtask workers do not, is first hand knowledge of the real-world context in which they are capturing photos. A microtask worker may see a photo in isolation and not be able to see an important detail or understand the scale, while a CDCV

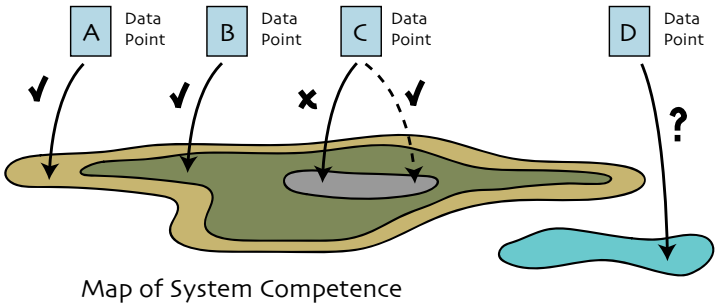


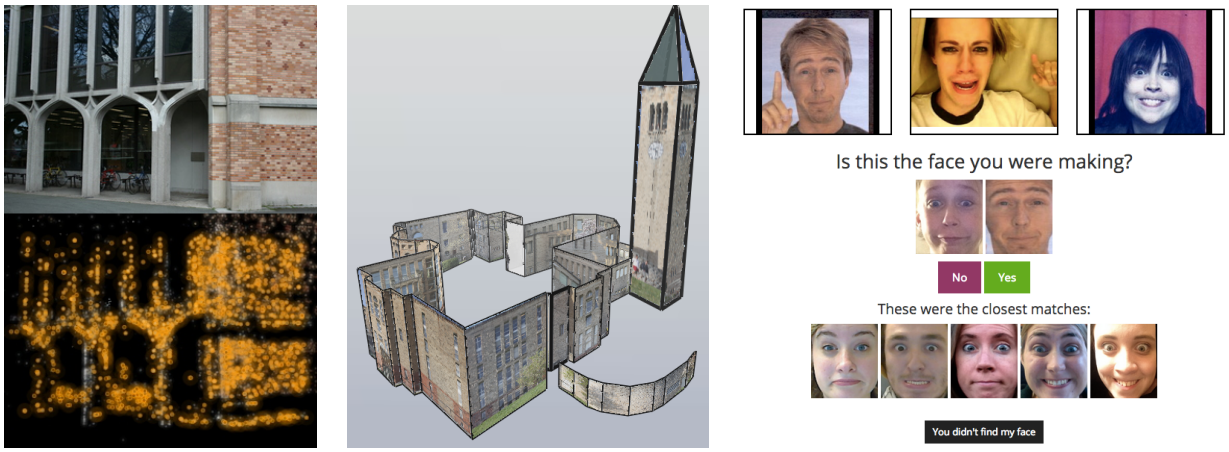
Figure 3.7: In addition to mapping new areas of a data space, interaction in a CDCV system can also take the form of evaluating the system's competence and validating the function that encodes new data into the map. This figure shows four new data points and how they each project onto the map. *A* maps correctly into the boundary, B maps correctly into an area of high system competence, *C* maps into an incorrect area, which a user could then correct, and *D* represents data not currently understood by the system. This interaction metaphor works especially well for CDCV systems performing inference (e.g., The Meme Quiz guessing expressions). Similarly, PhotoCity automatically predicts how each new photo registers with the existing point cloud, which it can get wrong.

user taking a photo has physical access to the real scene and can move around and even take additional photos to provide more information.

The final aspect of this property is that the interactions available to users allow them to make interesting choices. After understanding the map and which areas are in need of more data, users decide where exactly they wish to contribute. They also decide what strategies to employ (e.g., focus on a narrow region in depth, or go for breadth) and *how* they want to contribute. This freedom to make decisions about where and how to contribute is what makes CDCV systems *driven* by the crowd. Instead of top-down decisions being made about which region of a map is most important to work on, the users have the freedom to explore the regions they find most interesting. One reason for this is to motivate and engage users by connecting to their individual interests. But a bigger reason is that having a small group of people, or a powerful corporation, or an automated algorithm make decisions about what is important, is limiting and potentially damaging. Instead, a CDCV system allows many

people to understand a shared and likely complex technical system, and to collaborate by working on many parts in parallel and contributing unique perspectives.

3.1.4 Feedback In Response to User Contribution



(a) PhotoCity Feedback (b) PointCraft Texture Feedback (c) Meme Quiz Feedback

Figure 3.8: This figure shows three different ways CDCV systems provide feedback to help users understand the impact of their contributions. Figure 3.8(a) shows the user's photo next to a rendering of the point cloud that highlights the new points that photo added to the point cloud. The rendering is also from the same vantage point as the original photo, helping the user to understand if the photo was correctly registered. In addition to this visualization, the user also sees an overhead map view, number of points added, and which flags or castles were captured with this photo. Figure 3.8(b) shows some of the textures, computed from photographs, projected on the user-defined geometry. If the texture fits well, the user has defined geometry that is consistent with the visual information contained in the point cloud and registered photographs. In addition to this feedback, PointCraft users also get instantaneous feedback about how they are navigating in the world or using construction tool through sound and animation. Figure 3.8(c) shows the feedback presented to the user in the meme quiz, which includes the computer's guess and the five nearest faces. This shows the user how confident the computer is in its guess and how it might have gotten confused.

The fourth property is that the system provides feedback in response to a user's contribution related to how that contribution affects the ongoing mapping of the data space. Across my systems, feedback takes many different forms. In PhotoCity, there is feedback

for each individual photo that shows where on a map that photo was localized and how that photo contributed new 3D points to the building geometry. Feedback in The Meme Quiz is straightforward, consisting of the game's guess about which facial expression the user mimicked and a brief explanation of that guess in the form of images of the five nearest faces. In PointCraft, the feedback is more indirect; when a user creates geometry such as a plane, a texture is generated for that surface based on the input photos, which looks the most realistic when the user-defined geometry is the best fit for the point cloud. Each of these types of feedback are illustrated in Figure 3.8.

Feedback is essential for users to understand how their contributions affect the system, which then influences their future contributions. In the *HPU model* of labeling data, the previous three properties hold (although paid workers are rarely able to make interesting choices related to their interactive contributions), but workers generally do not receive direct feedback about their work. The only signals the workers receive are if they have been paid, blocked, or granted a bonus, or if they are in direct contact with the work requester outside of the AMT system. There is no evaluation available that can educate workers on where they are proficient and where they can improve, or where they are having the biggest impact.

When feedback is available, learning and improvement is possible. This can be a channel of motivation and engagement, especially when the interaction itself that yields feedback is intuitive. When a user has the choice of what to try, there is something to experiment with and if that experimentation leads to feedback that is comprehensible and interesting, that encourages repeated interaction. If done well, this can elicit *the SimCity effect* in which a system brings the user to an accurate understanding of the system's internal operations through playfully interacting and experimenting with that system [213, 214]. Learning is inherently fun, a theory explored by Koster's *Theory of Fun* [104], which is why my work has taken the form of games, as a way to frame learning about and mastering a complex system through interaction and feedback.

3.2 Outcomes

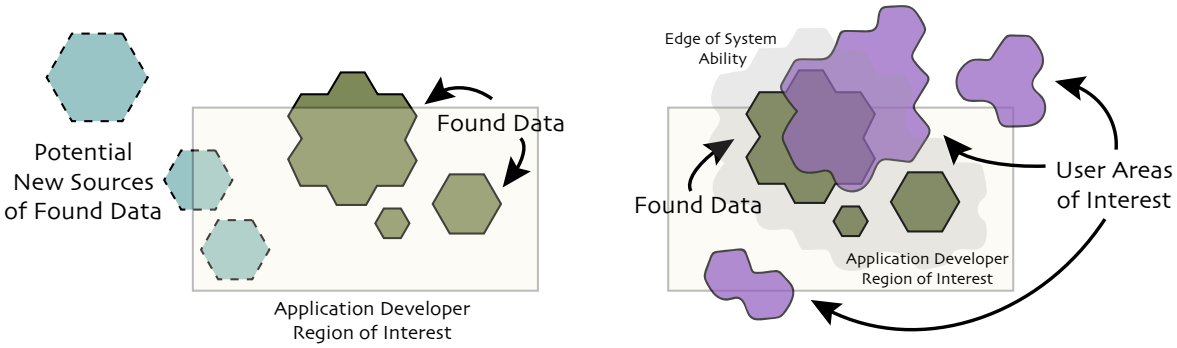
A crowd-driven computer vision system consists of the above properties, and when those properties interact, it can lead to the following outcomes: 1) data accumulates in areas of user interest, 2) new types of datasets can be collected, 3) it engages the creativity, skill, and diversity of the crowd, and 4) it motivates and focuses the effort of the crowd toward a shared purpose. These outcomes can, in turn, lead to more robust computer vision applications that are built on top of diverse and relevant datasets that are transparently and ethically collected, and drive innovation on a wider variety of problems that are of interest to end users. It is not necessarily the case that the properties above will certainly lead to these outcomes, but I have observed each of these outcomes in my work.

3.2.1 Accumulating data in areas of user interest

Because a CDCV system gives users a chance to experiment with inputting different kinds of data, and because they are free to make choices about what information they input into the system, users are likely to contribute data that is meaningful to them. Especially if they can see what other users have tried, or how the system has improved in response to others' contributions, users can make hypotheses around their interests and curiosities (e.g., will this building make a good reconstruction? Will my rendition of this facial expression be recognizable?) and put those hypotheses to the test. As a result of users contributing data in areas of their interest, the system will likely evolve to be more complete or more effective in those areas. Figure 3.9 illustrates how a system that organically grows in areas of user interest can overcome limitations of relying solely on found data.

3.2.2 Collecting new types of datasets

CDCV systems make it possible to collect new types of datasets, consisting of data that cannot be found or mined from an existing cache online or labeled by workers who only spend seconds on a task and lack useful context about the domain or receive feedback on



(a) Limited By Found Data

(b) Accumulating in Areas of User Interest

Figure 3.9: This figure shows how CDCV systems can overcome the limitations of found data by making it possible for data to accumulate in areas of user interest. On the left in Figure 3.9(a), an application developer may crawl the web and incorporate multiple sources of found data. Not all the data in a single source may be relevant, e.g., only a small fraction of photos tagged 'Rome' on Flickr show building exteriors, while many others show food, people, and interiors. To acquire more data, the application developer has to target *new* sources of data, which requires effort to locate, crawl, process, and clean, all while accounting for the idiosyncrasies of that new data source. Figure 3.9(b) on the right shows how areas of user interest can overlap with found data and region of interest of the application developer, but can also extend beyond to areas that are easily reachable by adding more data, or more distant and challenging. Additionally, the boundary of found data and a system's competence, as exposed through the map, interaction, and feedback of a CDCV system, can be its own source of user interest. Allowing users to organically grow the data in areas that interest them is a way to acquire data without having to *find* it, and can lead to data that is more tailored to the reality of how users use specific computer vision applications.

their work. CDCV systems allow for the accumulation of data that reflects the diversity and variability of users' perspectives and experiences, and in their interest in training computer vision systems that work for them and their needs.

Each of my systems generated a unique dataset. PhotoCity collected over 100,000 photos that provided dense coverage of building exteriors at two university campuses. PointCraft collected clean, simplified geometry paired with structure from motion output, which can be used directly or used to assist with learning to create clean geometry automatically. Whereas facial expression data is often focused on six basic expressions, or lacks labels for expressions

outside of those six, The Meme Quiz collected over 2,800 labeled facial expressions, captured in real world lighting conditions, of diverse and communicative expressions.

3.2.3 Engaging the creativity, skill, and diversity of the crowd

In a CDCV system, through the shared experience of mapping a data space, the contributions from one user can influence the choices and contributions of another user. This can foster inspiration and creativity, discussed in Section 2.3 and shown in Figure 3.10, which can affect engagement, as well as the quality of user contributions.

CDCV systems can take advantage of human skill in multiple ways. The interaction mechanisms of a particular system may enable the crowd to directly convey their high level human expertise, as in the case of PointCraft's interface for interactively tracing point clouds to fill in the holes. The feedback provided by a system may help the users understand the complex inner workings of that system and how their contributions in particular shape the system for themselves and others, as is the case with PhotoCity. The users also have direct access to the context in which they are capturing data, which can constitute its own form of expertise that remote, microtask labelers do not have access to.

Additionally, every person contributing to a CDCV system operates from her, his, or their own experience and provides a personal perspective, and CDCV systems are designed to be open to a variety of inputs without being too prescriptive. The users, their interests, their curiosity, and their choices are what drive the system.

3.2.4 Motivating and focusing crowd effort toward a shared purpose

The transparency into the underlying purpose and agency afforded in contributing to that purpose can be a motivating factor for users to engage with CDCV systems. Over half of people surveyed about their participation in PhotoCity stated that the purpose of reconstructing 3D models from photographs was a reason for initially trying out the game [198]. Competition was another common motivator for players, which was possible through the

feedback mechanism which awarded game points that were directly related to the photos and geometry contributed by players.

3.3 Conclusion

This chapter presented the *crowd-driven computer vision* framework, comprised of four properties that must be present in a system to make it a CDCV system. Those four properties are as follows:

1. There is a specific computer vision technology identified and the purpose of the CDCV system is to collect data to be used with that technology

2. There is a map (literal or conceptual) of the space of data being collected and it is meaningful to think about continuing to map that data space

3. The user interaction of the CDCV system empowers users to collect and contribute data

4. The system provides feedback in response to users' contributions, which explain to the user the effects of their contributions

I claim that these properties are necessary for a system to be a crowd-driven computer vision system, and that because of the ways these properties interact with one another, the combination will likely yield (but is not guaranteed to cause) the following CDCV outcomes:

1. Accumulating data in areas of user interest

2. Collecting new types of datasets

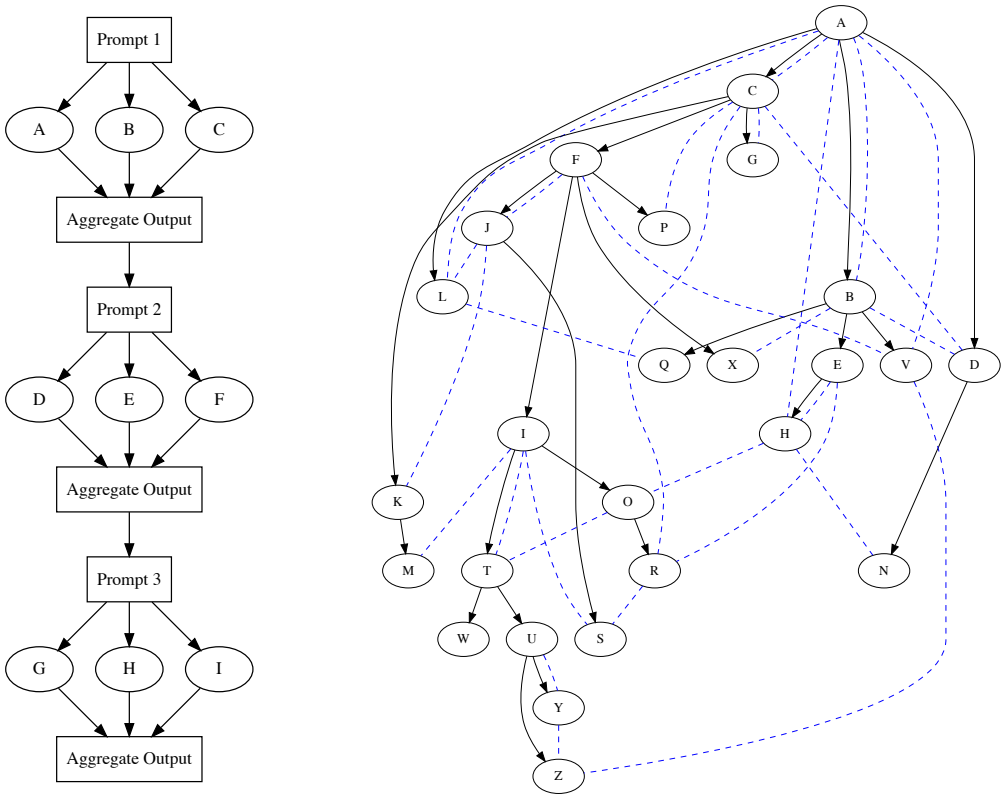
3. Engaging the creativity, skill, and diversity of the crowd

4. Motivating and focusing crowd effort towards a shared purpose

Crowd-Driven Computer Vision	
Properties	**Outcomes**
Core CV Technology	Data Accumulates in Areas of User Interest
Mappable Data Space	Can Collect New Types of Datasets
Interaction	Engages Creativity, Skill, and Diversity of Crowd
Feedback	Motivates and Focuses Crowd Towards Shared Purpose

Table 3.1: Summary of CDCV properties and outcomes. The presence of all of these properties in a CDCV system will likely yield the outcomes on the right by making the data space of a specific computer vision system interactively mappable, and by supporting crowd collaboration.

The three systems presented in this dissertation, PhotoCity, PointCraft, and The Meme Quiz, are representative of these properties and outcomes, each in its own way. The following chapters go into depth in each system. The focus of Chapter 4 is how PhotoCity players learn about the underlying computer vision system through their interactions and the feedback they receive, and are able to develop strategies based on their learned intuition of the system, and to collect a large amount of highly relevant data in a limited time. The focus of Chapter 5 is on the interactions that PointCraft provides to directly identify and fill in gaps and ambiguous areas of the spatial map of a noisy, partial 3D point cloud. In Chapter 6, the space being mapped is more abstract than in PhotoCity or PointCraft, as it represents the high-dimensional space of facial appearances, facial expressions, and capture conditions. The focus of the Meme Quiz chapter is on creating an interaction loop that is much shorter (a few seconds versus spending hours or days in PhotoCity or PointCraft) and requires less skill to drive an interactive computer vision system that updates its behavior with every subsequent input from the crowd.



(a) Structured, isolated task pipeline

(b) Deep chains of inspiration and remixing

Figure 3.10: This figure shows two types of complex crowd work. On the left, Figure 3.10(a) depicts a rigid task pipeline (the HPU model) that might be deployed on Mechanical Turk. At each stage, several workers work on the same prompt to produce an output (represented by a letter), and then those outputs are used together through, e.g., through independent agreement or voting, to produce another output that may be the input to the next stage in the pipeline. Work is done in isolation, which may be necessary for that type of task, but it leaves no room for workers to draw inspiration or learn new techniques from their coworkers. On the right, Figure 3.10(b) shows an organic structure for crowd work that fosters creativity, in which contributors can see and be inspired by each other's outputs (blue dashed lines) and can directly remix or build on the work of others (black solid lines).

Chapter 4

PHOTOCITY

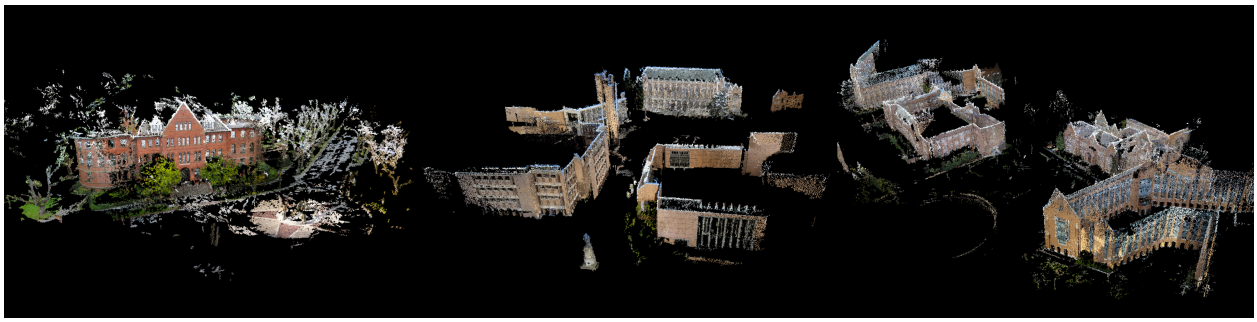


Figure 4.1: Point cloud models of University of Washington campus reconstructed from thousands of PhotoCity photos

The first full example of a CDCV system is PhotoCity [197, 198]. This chapter explores how the system design and interaction of the game supports players' objectives of mapping their school campuses through photos and how the feedback mechanism in particular helps players develop expert strategies that represent their intuition of the underlying computer vision system, developed solely through interacting with the game. In addition, the players of PhotoCity collected a large-scale structure-from-motion dataset consisting of over 100,000 highly relevant photos providing dense coverage of the area.

4.1 Introduction

Imagine having a huge collection of georegistered photos taken from every corner of a city, inside and out, at many different times of day, different seasons, across many years. Such a photo collection would have many applications. You could give your faraway friend a tour, or scope out a new part of town before going there. For any location, you could see it during

the day, at night, during summer or winter, and from any angle. If a building had been torn down or rebuilt, a snapshot of how it once was would be preserved. These localized photos could be used to enable dozens of new augmented reality applications, could help in monitoring and analyzing building infrastructure, and could aid in urban planning and analysis of changes in cities over time. The photos could even be used to build a detailed 3D model of the entire city using computer vision techniques [181, 1].

While large sets of urban photos exist in very structured (e.g., Google Maps and Street View) as well as unstructured (Flickr, Facebook, Picasa) collections, these existing collections have a number of drawbacks in the context of such mapping and analysis applications. The structured collections (usually captured by commercial enterprises, most famously Google) are quite expensive to acquire, only available in certain areas, and typically capture only certain areas such as city streets and popular footpaths. These structured collections tend to be "refreshed" infrequently and thus photos may often be out of date; similarly, only a single time of day or year is usually available. On the other hand, unstructured collections (e.g., the millions of photos of Rome on Flickr) are captured "for free" by thousands of people (many of them tourists) walking around every day with cameras. However, only landmarks, such as the Colosseum and Trevi Fountain, are well-represented, and most of the rest of the world is very sparsely photographed. For instance, in recent work in computer vision, 150,000 Flickr photos of Rome were downloaded and automatically reconstructed into a set of 3D models spanning the city [1], but these models only represent disconnected pockets of 3D structure corresponding to famous landmarks. Less popular areas such as side streets were not photographed enough times to be modeled.

What if there were a way to effortlessly and cheaply collect the vast number of 'missing' photos—without special equipment and with the effort distributed among millions of photographers in the community? We have developed PhotoCity [197, 198] to crowdsource the gathering of these photos in a fun and effective way, focusing on the application of 3D city modeling. PhotoCity is a game played by simply taking photos. The game processes the captured photos using computer vision techniques and uses them to incrementally expand

3D models; players 'capture' virtual flags and castles as they contribute to digital 3D replicas of real buildings. The result of the game is a set of detailed 3D models and a large set of photos that densely cover an entire area from many different angles. A preliminary version of the game is described in [197], with attention on the design of a game that enables players to collaboratively construct 3D models by taking photos. In this chapter, we apply PhotoCity to the real task of modeling two university campuses, demonstrating what the game can accomplish with a small group of expert players who have been trained by playing the game. We seek to determine whether the game dynamics we designed for PhotoCity are conducive to large-scale distributed data collection. We present the detailed results of a field study to determine the effectiveness of our game PhotoCity, as well as present design changes made as we received feedback from this experiment.

PhotoCity, like other games with a purpose (GWAPs), provides incentives and game mechanics that drive people to perform a useful task, in this case, to capture missing photos needed for 3D modeling. However, a unique challenge in PhotoCity is to find ways to take people's normal behavior when taking photos—i.e., to take a few well-composed shots from "canonical" viewpoints of the fronts of buildings—and change them so that they take a larger, more useful set of photos for the task at hand. PhotoCity addresses this through a map-based user interface design and competitive game mechanics, but we still found that it often takes some time for users to warm up to how to play the game. To be a viable mechanism for large-scale data collection, the game must engage the interest of enough players for long enough for them to "learn" this new photo-taking behavior and collect a sufficiently complete set of photos.

To evaluate whether PhotoCity is an effective way to collect a comprehensive set of photos of a large area, we held a competition between two universities. The competition brought in over 100,000 photos; compared to photos of the same areas available on Flickr, the PhotoCity collection provides much greater geographical coverage, and is much more effective for 3D reconstruction.

These 100,000 photos were contributed by a relatively small number of players (forty-

five). Despite this small number, the game was engaging enough for these players to capture a very sizable collection of useful data, and the results suggest that crowdsourcing is a viable mechanism for large-scale acquisition of visual data.

In this chapter, we will briefly outline the basic mechanics of PhotoCity and the ways in which we guide players' actions towards the purpose of the game. We will then describe enhancements made to the game along the way to focus player efforts (on completing buildings) and increase player engagement (by organizing a competition between two universities). Finally, we will discuss the outcome of our university competition: that the success of PhotoCity depends on the participation of highly-motivated expert players who become so involved in the game that they play for an extended period of time, are able to hone their skills, and contribute thousands of photos each.

A key contribution of this chapter is the design and evaluation of a game for large-scale, targeted data collection. An overarching lesson is that the success of such a game is not necessarily dependent on the number of players. What is most important is focusing player effort and providing competitive or exciting situations that lure people in long enough to turn them into expert players.

4.2 Related Work

This section discusses related work in games with a purpose (GWAPs), collaborative tools for mapping and 3D modeling, and 3D reconstruction via computer vision.

4.2.1 GWAPs

A game with a purpose (GWAP) is a means of extracting useful output from a human player in a game setting in a way that is fun for the player. Many GWAPs are designed to have humans perform tasks that are difficult for computers, such as labeling images and folding proteins. An in-depth examination of relevant GWAPs is covered in Chapter 2. In contrast to traditional GWAPs, ours leverages humans for the physical activity of going outside and taking a picture (or set of pictures) in a specific place.

4.2.2 Collaborative mapping and modeling tools

A number of recent systems have utilized collaboration from large numbers of people to generate large-scale public goods related to online maps (following an altruistic incentive model, like Wikipedia [1]). Most relevant to our work is OpenStreetMap [75], an online system that allows people to contribute GPS traces and edit maps in the service of creating a open map of the entire world, complete with street networks, parks, building outlines, and many other features. Our work is complementary, seeking to supplement such maps with dense sets of ground-level photos contributed by the community (as well as 3D models resulting from processing these photos with computer vision techniques). A related project, Wikimapia [69], allows users to draw overlays and annotate regions on aerial photos, resulting in more semantically rich maps; the LabelMe system [161] provides similar functionality for Internet photos. Google has also released several tools for individuals to create 3D models of real-world buildings, including SketchUp [28] and Building Maker [120]; the goal is to use these models to populate city maps. These tools have grown out of work on interactive modeling in the CAD and computer graphics communities [223, 43, 174, 138]. Unlike all of these systems, our work explores the use of competition in a game setting to provide incentives for users to contribute content.

4.2.3 3D Reconstruction

PhotoCity is designed around the goal of acquiring photos for use in 3D modeling, and uses as a back-end recent work in computer vision for automatic 3D reconstruction from 2D images. This computer vision technology used stems directly from the work of Snavely et al. on Photo Tourism [181] (also incorporated into Microsoft's Photosynth), as well as the multi-view stereo work of Furukawa and Ponce [66]. This technology takes a set of photos and automatically reconstructs the viewpoint (position and orientation) from which each photo was taken, as well as a "point cloud" representation of the scene; example 3D point clouds

[1] `https://www.wikipedia.org/`

are shown in Figures 4.4, 4.5 and 4.9. PhotoCity exposes these models to players through a map-based interface, one for each building in play, and actively encourages players to capture views that extend and expand these models. Unlike Photosynth, where each model is built by a single user, PhotoCity players collaborate to reconstruct the world. These computer vision techniques can also be run on large collections of user-contributed photos on sites like Flickr. However, PhotoCity encourages *new* photos to be taken, whereas the tourist photos on photo-sharing sites are largely concentrated around a small number of viewpoints and landmarks. While this data is extremely useful for understanding and visualizing where people take photos and reconstructing famous landmarks [171, 95, 152, 37], in our context we seek a much more complete set of photos for our task. Even extremely large sets of Flickr photos are insufficient for reconstructing *complete* models of buildings and cities [1, 60].

In contrast to scraping the web for images, the Wiki-based city modeling approach [84] of Irschara et al. solicits photos from the users in much the same way as PhotoCity. By contributing their photos to a central repository, wiki users (and PhotoCity players) are helping to build a 3D model of a location of shared interest. The difference is that PhotoCity provides additional incentives for players to upload their photos, such as capturing virtual flags and scoring points.

4.3 Game Description

A description of a preliminary version of PhotoCity and the underlying computer vision technology is given in [197], but we recap and expand on it here. The core mechanic of our game involves players inspecting the state of the game world on a map, taking photos at locations of promising in-game value, uploading the photos to the PhotoCity website, and then observing the results of their play. Through repeated cycles of this process, summarized in Figure 4.2, players introduce new photos of the game world into play, which in turn expand the 3D "point cloud" models stored on the servers. The geometric points in these point clouds correspond to points on the surfaces of buildings, trees, and other objects, and also to the game "points" that players accumulate through the course of the game.

4.3.1 The Game World

The game world of PhotoCity is a map of the real world overlaid with virtual castles and virtual flags, as shown in Figure 4.3. By contributing photos in certain locations, players can capture and control these castles and flags. Each castle corresponds to one partially reconstructed 3D model of a real-world building. If a player adds the most points to the school library, for example, that player "captures" the building, and his name appears next to the library's castle icon on the map. Flags correspond to specific map locations in the real world, and are automatically placed along building walls to guide players to more lucrative, missing viewpoints. A player controls a flag when her photos contribute the most points to that one section of wall. As models grow larger, more flags appear along the edges of the model.

4.3.2 Basics: Earning Points and Capturing Flags

Every virtual model in the game is represented as a dense point cloud, like the one pictured in Figure 4.5. When a player adds a photo to a model, our back-end server estimates the 3D pose of that photo with respect to the current 3D model, then matches pixels in that photo to existing, nearby photos, in order to triangulate new points—in essence, each photo acts similar to a "laser scan" captured at a particular viewpoint, with geometry created by matching points in that photo to other views. This process literally adds new 3D points to the model, and the player earns one game point for each new 3D point.

To play the game, the typical player looks at the map of the game, identifies the flags she wishes to capture, and then takes photos of that portion of the building. For a photo to earn points, it must (1) overlap with enough existing points in the model and (2) overlap the empty space next to a model. The first requirement allows the photo to connect with the model and have its position within the model automatically calculated. The second requirement, that the photo look beyond the existing model, allows new points to be added to that void as soon as there are enough other photos to triangulate the 3D positions of those

Figure 4.2: Gameplay cycle: players look for flags to capture on the map, go outside and take photos, upload the photos, and then capture flags and conquer buildings.

features. A photo can add up to several thousand new points.

Instead of taking a nice photo of the front of a particular building, an area that is likely already saturated in photos, the player can earn many more points (and capture flags) if he goes to the *edge* of the virtual model and takes entirely *new* photos there. This basic game mechanic accomplishes two things towards the purpose of PhotoCity: (1) Photos are not just of the 'popular' locations, but include many different locations and many different viewing angles of each location. (2) Every photo that a player earns points for has already been determined by the game to be useful.

A verification mechanism is built into the game. Every photo is compared against a 3D model when it is uploaded, and the only photos that earn points are photos that overlap with or fit into an existing model. If a player uploads an irrelevant photo, of a different building or of a flower for example, that photo will fail to match the 3D model and the game will mark it as 'unsuccessful'. The game performs this matching in a matter of seconds and then the player views which photos failed and which worked. They can use this information to modify their approach to taking pictures. Occasionally, a photo of of the correct building fails to match because the virtual model of that building is not yet large enough to incorporate that photo. The system automatically retries each photo three times, but if a photo repeatedly fails to match (because the model has not grown enough), players are free to re-upload photos. When they do, the game considers these as entirely new photos.

4.3.3 Advanced Technique: Seeds

In addition to playing PhotoCity by growing existing models, players can also seed their own models.

Every model in PhotoCity starts off as a seed generated from a small batch of photos of the real building. The number of photos used to make a seed is between 20 and 200. Models in their starting state usually only span one face or one corner of a building and have rough edges and large holes where data for the building has yet to be captured. Figure 4.4 shows the size of a seed made from thirty photos. Once a seed has been generated, it is aligned

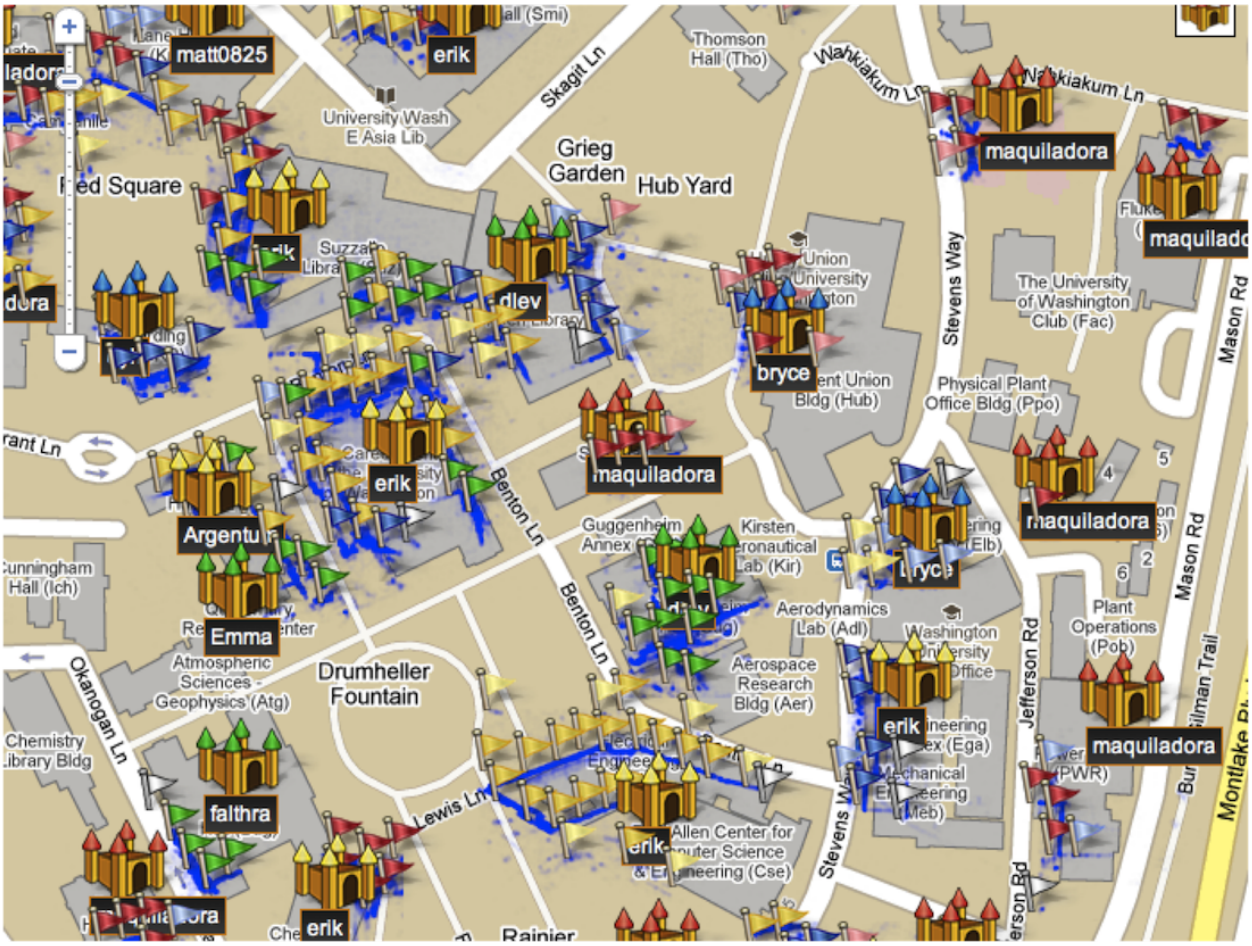


Figure 4.3: PhotoCity map with castles and flags at University of Washington. Different teams (at the same school) have different colored flags and castles. When a player captures a flag, the flag becomes the color of her team. When a player captures a building, her name appears under the castle icon.

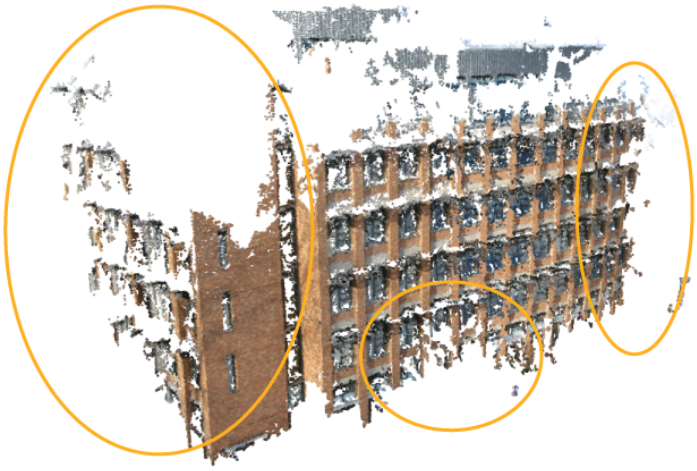


Figure 4.4: A seed made from 30 photos. Players take photos to fill in the missing holes (indicated in this image by the ovals) and to expand this seed from a single facade to the entire building.

with the map and added to the set of active, under construction models in the game.

If a player wants to take pictures of a particular building but that building is not already in the game, that player can seed that building herself. To do so, she takes a starting set of photos that capture just one side or corner of a building from all possible angles. Then she uploads those photos through a separate web interface and waits for the game to generate the model using the techniques of [181] and [66]. This can take up to several hours, depending on the number of photos. It can also fail to work if the building is too shiny or featureless (e.g. a building with large glass windows). If the seed generation was successful, the player then aligns an overhead view of the model with a map and waits for the game developers to approve her seed and her alignment. We make sure that each seed meets a minimum quality requirement and is neither offensive nor irrelevant.

Letting players start seeds themselves is beneficial to the game's purpose and to the player. It is impossible for the game designers to seed a large number of buildings, especially if they are not physically present where PhotoCity is taking place. But it is easy for players to start seeds anywhere and everywhere, and earn many points for each seed. Players can choose

buildings that have personal meaning to them or are convenient to photograph. Having so many models active in the game gives other players variety for where to go and take photos.

4.3.4 PhotoCity-style Photography

The photography style required in PhotoCity is different from ordinary photography. Van House [202] examines the types of photos posted on Flickr and finds that most represent a way of 'life chronicling' and self-representation. Baber et. al. [7] point out that "much 'tourist photography' represents a special form of image capture" in which tourists tend to gravitate towards the best vantage points to take their own versions of photos seen in brochures. PhotoCity requires a more utilitarian approach to photography than the artistic approach seen on Flickr, but also requires more creativity than normal 'tourist photography' to get away from canonical views and capture a scene from entirely new angles.

Instead of carefully composing a single shot, the key to PhotoCity is to take as many photos as possible from many different angles. The ideas of quantity over quality, and sweeping the camera around a scene, are probably the most radical notions that people new to PhotoCity have to learn. Figures 4.5 and 4.12 show camera/photographer positions and how players walked around a particular model taking photos every few steps. This is but one dimension on which to change how players take photos, favoring quantity and variety over nice composition. In the end, the photos taken for PhotoCity should have the coverage, density, and variety to essentially blanket an entire location in photographs.

4.4 Field Study

In our setting, the main challenge is to acquire a different and much larger set of photos than what people normally take and post online. This set of photos should also be large enough and comprehensive enough to be completely "cover" a target area (and in particular yield complete 3D reconstructions of buildings). This could be accomplished either by having many players stray slightly from their ordinary behavior and do a small amount of work, or having fewer players drastically change their photographic style and each contribute a large

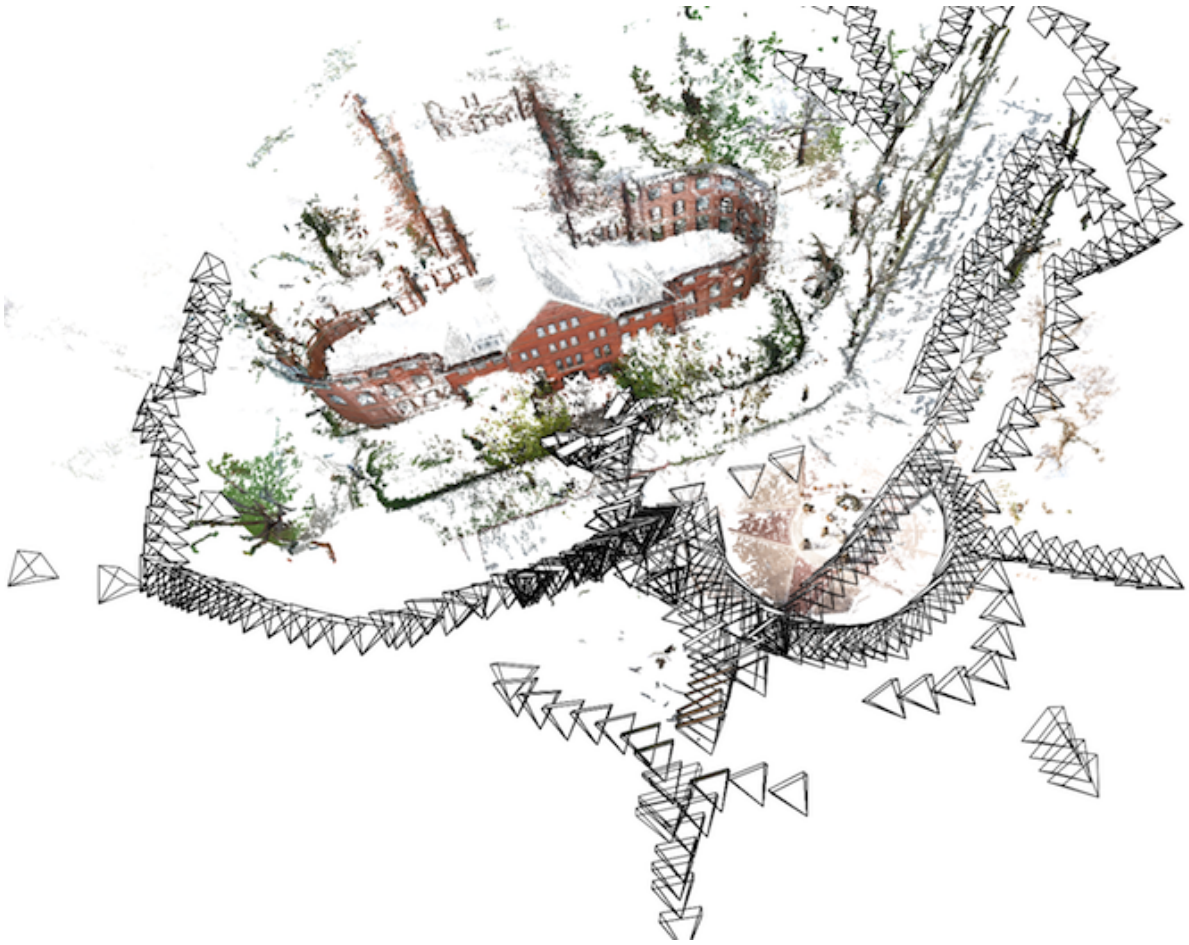


Figure 4.5: A near-complete model with camera positions shown as black triangles around the model. The game requires a different style of photography, one that favors quantity and variety over artistic composition. In order to expand a model, players must take photos that overlap the existing model, and then move to a new area, taking many photos along the way.

amount.

We designed PhotoCity to have a variety of different incentives, and deployed it as a competition against two universities to evaluate its success and find whether the game solicits many new users, a few passionate users, or something in between.

4.4.1 New Incentive: Competition

In the Spring of 2010 we instigated a rivalry between two schools. For six weeks, split into two three-week rounds, Cornell University and the University of Washington in Seattle played PhotoCity and competed to see which school could reconstruct the "best" model of its campus. Players could view the PhotoCity map for each school, or track the game on a single competition webpage. Anyone at either school could sign up online and start contributing photos. We advertised though school newspapers, department mailing lists, and temporarily ran an advertisement on Facebook targeted at students interested in photography.

The competition was not just between schools, but also between players (within and across schools). The competition page featured a leader board where players were ranked by number of points and by number of successful photos. There were also five 'titles' that one player at each school could hold: Kingdom Overlord (owning the most models), Expert Expander (spawning the most new flags), Expert Seeder (starting the most new seeds), Master Flag Conqueror (most flags captured), and Best Recruiter (players could invite their friends to play and get recruiter credit). Finally, within each school there was also a team competition; players could select from four teams (red, green, blue, yellow) to join, and the webpage provided a ranking of each team by score.

We awarded prizes after the competition: T-shirts for the top 15 players (based on points), Flickr Pro accounts for the title-holders at each school, and 3D laser-etched trophies of models built during the game for the top player at each school.

(a) (b)

Figure 4.6: Collectable PhotoCity gems entice players to complete buildings. 4.6(a) shows gems on the unfinished side of a model while 4.6(b) shows gems on two completed models.

4.4.2 New Mechanic: Gems

The competition took place in two rounds. The first round lasted three weeks and players played with the game mechanics described above, seeding and expanding models. We discovered at the end of the first round that while players seeded many buildings throughout each campus, most reconstructions were not complete.

The theme of the basic PhotoCity mechanics, and of holding the competition, is to focus player efforts to get something productive done. For the second round, we expanded on this theme by adding a new mechanic designed to entice players to complete buildings.

This new mechanic was collectable gems. Gems are very similar to flags in that they represent specific locations on the map. But unlike flags, which are automatically placed

on existing building walls, gems are manually placed (by the designers) on the far sides of buildings. Unlike flags, which different players can contribute points to, capture, and steal from each other, gems are *collectable*. The first player to grow a model into the vicinity of a gem collects that gem and no other player can steal it from the first. Figure 4.6(a) shows a building with a seed started on one side of the building and gems placed around the far side. Gems turn from gold to purple when they are collected. Figure 4.6(b) shows two buildings that were completed during the second round with the gem mechanic.

4.5 Results

In total, forty-five players played PhotoCity for the six weeks of the competition and submitted over 100,000 photos. There were 26 students from University of Washington and 19 students from Cornell who participated, mostly recruited through department mailing lists and word-of-mouth. Figure 4.10 shows the number of players who were "active" and submitting photos on any given day, while Figure 4.8 shows the total number of photos submitted on each day of the competition. Activity and enthusiasm at both schools was roughly equal, though Figure 4.7 shows Cornell maintaining a lead in the number of points for most of the first round, despite having fewer players.

The first round started with four seeds at each zone, and by the end, players had started 64 seeds at University of Washington and 55 seeds at Cornell, totaling 119 new seeds. Figure 4.12 gives an idea of the distribution of seeds at the University of Washington. The top two seeder players began 26 and 20 seeds respectively.

During the first round alone, players submitted over 76,000 photos. During the entire competition, players submitted over 109,000 photos. Players were able to re-upload photos that initially failed to match, and 31% of the final photo count were resubmissions. About 68,000 photos of those photos registered, amounting to about 500 photos on average per model. Three completed models from the game are shown in Figure 4.9. We now compare our photo set against Flickr to see if the game accomplishes its goal.

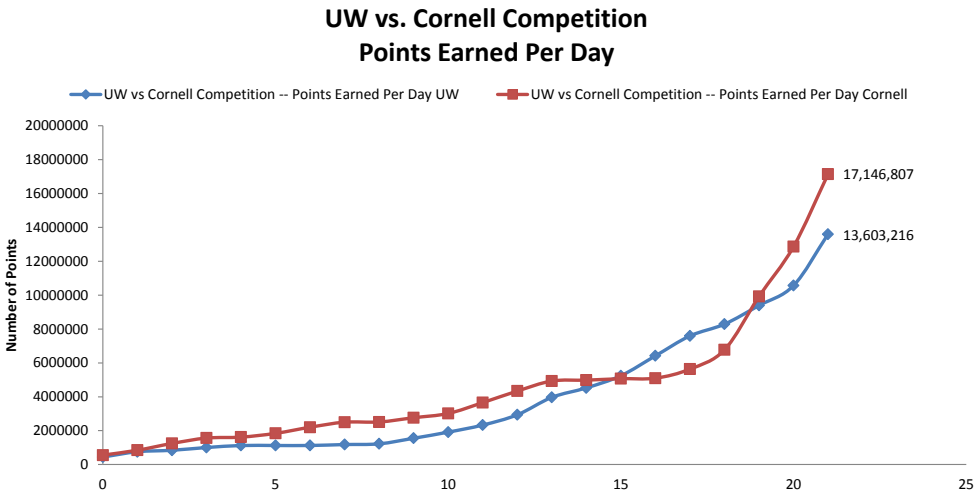


Figure 4.7: A plot of the points earned by UW and Cornell over the first round of the competition, demonstrating that competition made a significant impact in effort exerted by players. Cornell's points are shown in red and UW's points are shown in blue. Cornell held the lead for the first two weeks, and when threatened by UW at the start of the third week, defended their first place position.

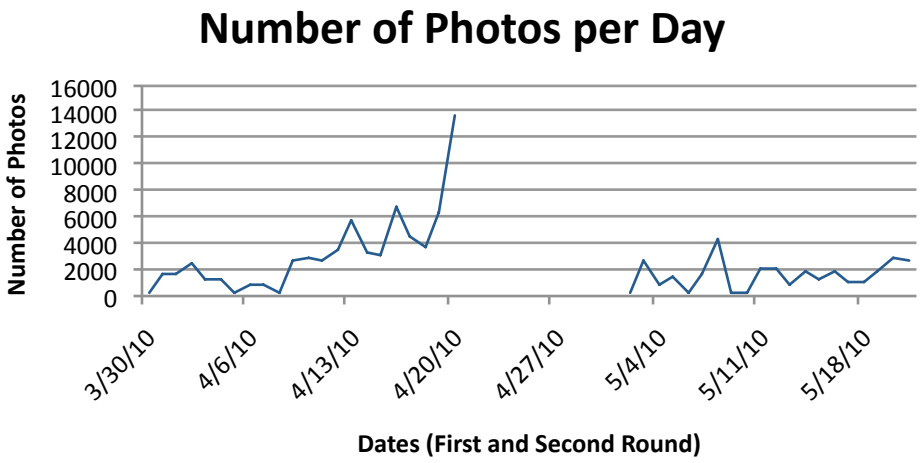


Figure 4.8: This graph shows the total number of photos submitted per day by students at both schools during the first and second rounds of the competition, each of which lasted for three weeks. Daily photo-submission is higher in the first round, especially towards the end, because there are over 50 active buildings per campus instead of the 10 active buildings per campus in the second "gem-collecting" round.

4.5.1 Versus Flickr Photos

PhotoCity collected 109,000 photos divided between University of Washington and Cornell. In contrast, the Rome in a Day project used 150,000 photos in its Rome reconstruction. We totaled up the photos of each of the connected components of Rome (the largest of which contained 2,106 photos) and calculated that around 10% of the 150,000 photos were used in any 3D model. The rest do not usefully contribute because they are of people, interiors, or other kinds of photos meant to provide some personal value to the photographer.

In contrast, 60% of PhotoCity photos are used! Furthermore, while some of the unused photos are definitely unrelated, many are still likely to be of school buildings. They did not overlap a model when they were added, but may have overlapped at the end of the game if retried.

Again compared to Rome, the largest connected reconstruction has 2,106 photos in it. Our largest single model is of the Cornell Arts Quad with over 4,000 photos and 8,000,000 points. Each campus reconstruction consists of several multi-thousand photo/multi-building reconstructions.

We found PhotoCity photos to be significantly more useful for 3D reconstruction than Flickr photos of a popularly photographed city of Rome. Of a less popularly photographed location such as a university campus, this difference was even more striking. Figure 4.11 shows 35,000 PhotoCity photos densely packed across University of Washington campus and there are only about 30,000 geotagged photos on Flickr, most of which are not suitable for modeling the campus buildings.

Even though there were only 45 players, these players took thousands of photos of *views that had never been captured before* and covered their campuses much more thoroughly than an unorganized effort could reasonably achieve. Another round of PhotoCity could fill in the remaining gaps or even chronicle the campus changing over time. This conclusively demonstrates that PhotoCity is an effective way to collect large, dense photographs of places previously not thoroughly photographed.

4.6 Discussion

The results of our experiment show that it only took a small group of players to capture a large amount of data. We now discuss what motivated these players to participate so enthusiastically, how they became so effective, and what positive outcomes they experienced personally.

4.6.1 Player Motivation

At the end of the competition, we presented players with a survey that included questions about what motivated them to start playing and keep playing, and what strategies they developed to gain a competitive advantage. Twenty-two players responded. While this survey mainly captured the opinions of those who played through the end of the competition, the responses were in line with what we observed during the study: that competition was a good motivating factor and that most people who played were also interested in the resulting 3D models.

There were different types of competition: competition between schools, rivalries between players, and players trying to climb to the top of the leader board or earn one of the five titles (e.g. "Expert Seeder"). The graph in Figure 4.7 shows evidence of the competition between schools having an impact on play. Cornell, in red, held the lead for the first two weeks of the competition. When University of Washington temporarily surpassed Cornell, Cornell quickly stole back its lead. In talking to students at both schools, Cornell students were more interested in winning the competition, while University of Washington students were interested in competing among themselves. Most of the top players at the University of Washington were friends and spent their time trying to outdo each other.

Players also cared about the game's purpose. More than half of survey responders cited it as a reason for trying PhotoCity initially. Unlike Photosynth, PhotoCity produces dense, detailed 3D models of real world locations and allows for people to collaborate and improve

models. The popularity of Photosynth (over 400,000 synths created in the first year[2]) suggests that people want to thoroughly capture the meaningful things around them in photos.

The main de-motivator of players, especially novice players who did not submit very many photos to begin with, was when their photos were rejected by the system. Observing Figure 4.13, we suspect players have to take more than 10 or 20 photos to reap benefits like seeing a model actually grow, which make the game fun.

4.6.2 Players Becoming Experts

The majority of point contributions (80%) during the university competition came from the top 10 players. The top player alone contributed 20% of the total number of points. PhotoCity appears to be a game for experts, with the best players taking significantly more photos and earning significantly more points than novice players.

The difficulty for new players stems from the need to adopt a different photography style and to learn (from the map) what to take photos of. Most of the players who end up playing PhotoCity a lot start off with a reasonable idea of how to play (take lots of photos) and then get even better with experience. The top line in Figure 4.13 shows players who played 10 or more days taking gradually more photos each day over the first few days. The bottom line in the same figure shows players who only played for one or two days taking very few photos to start out with, and then apparently being so discouraged or uninterested that they took even fewer photos the next day. A possible improvement to the game is to encourage a new player to take many photos (at least 20, though 50 or 100 might be better) right away, (a) so that they can make a sizable contribution, and (b) so that they can earn a large number of points right away and feel inspired to play again.

---

[2] `http://blogs.msdn.com/b/photosynth/archive/2009/08/18/photosynth-turns-one-year-old.aspx`

4.6.3 Expert Strategies

Further evidence of players' deep involvement in the game is the strategies they described to us in the survey. Different strategies involved playing with a different game mechanic, finding the most lucrative types of buildings, or optimizing for time of day, but each was developed to give the player a competitive advantage.

Regarding what types of buildings were most lucrative:

> "Targetting buildings without huge slabs of glass/windows works best. Buildings that have plain facades are tough to get points on. Buildings with stone exteriors, especially rough hewn stone, have tonnes of points. Brick buildings can give lots of points if the grout is especially thick."

Players mentioned that the corners of buildings and trees planted next to buildings were especially difficult to navigate around. One player described how to tackle corners, while another mentioned an alternate strategy for earning points that avoided dealing with corners.

> "I've been trying to work out my corner strategy, and mostly it consists of taking a lot of pictures of that corner from a lot of angles."

> "I thought that since it seemed to be very hard to mesh around corners, I would make lots of seeds and try to get points that way."

Some strategies optimized over lighting and time of day.

> "The third strategy was to try and approximate the time of day and the lighting that some pictures were taken; this would help prevent rejection by the game. With that said, cloudy days were the best by far when dealing with corners because less contrast helped the game figure out the geometry from my pictures."

> "When it was sunny, I utilized my camera's anti-shake features to allow myself to take photos while walking around a particular zone. This allowed me to take crisp photos while not having to stop my stride. Also bonus: less people stared at me!"

Finally, several of the players mentioned how they planned their photo-taking sessions and kept their photos organized.

> "Taking many pictures of one building with 5 megapixel camera, then coming home to submit, sometimes up to 5 gigs on a weekend shoot, was primary photo capture strategy. Organizing photos on external hard drive folders by campus region, building, then face of building provided good hierarchy to structure uploads."

> "I used my ipod while on campus to help me pin-point uncaptured flags. I would then write out a checklist of places to go to take photos (kind of like a photo-route) If unsure, I would just check it again. It was pretty nifty."

4.6.4 Positive Outcomes for Players

Although the primary purpose of PhotoCity is to collect photos, players experienced personal benefits through the game play. In the survey, we asked players whether they felt they got to know their campus better. Twelve of the 22 participants surveyed said yes. We also asked whether they took photos as part of their daily routine or went out of their way to take photos. Thirteen people responded, 'I went out of my way to take photos,' eight said, ' I didn't stray far from my normal route, but did seek out new vantage points,' and only one participant responded with 'I only took photos where I was already going.' We can conclude that PhotoCity made many of the participants more physically active than their default state.

Only a small group of players participated, but they still contributed a surprising amount of data. Given time to gain expertise and incentives to become deeply involved in the game, players can become extremely effective.

4.7 CDCV Properties and Outcomes in PhotoCity

PhotoCity clearly exemplifies each of the properties CDCV. Structure from motion (SfM) is the core computer vision system, and the game is built around a modified version of the Bundler SfM software [179]. The geographic map that shows where data currently exists and where the boundaries of that data are is what drives the core mechanic of the game. The interaction made available by the game is that of taking batches of photos and uploading them to certain regions on the map. The feedback that the system provides to the player directly represents how the player's contributions have changed the map.

PhotoCity also clearly demonstrates each outcome of CDCV, from accumulating data in areas of user interest (e.g., through seed models), in creating new types of datasets (e.g., datasets that provide more dense coverage than what can be found online), in engaging the skill and creativity of the crowd (e.g., players developing strategies and even finding ways to 'cheat' that are beneficial for the underlying purpose), and motivating and focusing crowd effort (e.g., coordinating and incentivizing 45 people to collect over 100,000 photos).

While PhotoCity is a good representation of the design concepts of the CDCV framework, each CDCV system property is deeply tied to the specific 3D reconstruction purpose and it may be hard to directly map patterns from PhotoCity onto a new application. In contrast, The Meme Quiz (discussed in Chapter 6) is a lighter-weight system that may be more easily adapted to new scenarios.

4.8 Conclusion

The problem we set out to address was to gather enough photos of a certain location, such as a university campus, to be able to thoroughly reconstruct that location in 3D. We have demonstrated that our game, PhotoCity, is capable of bringing in many thousands of highly

relevant photos–photos of buildings from a diverse set of angles and viewpoints. During a competition between the University of Washington and Cornell, players took over 100,000 photos, over 60% of which were used in the resulting 3D models. It only took 45 players to collect this data. To achieve such success with a small number of players, we designed various game mechanics that made players' in-game actions productive and beneficial to our purpose, and motivated our players through competitions, prizes, and a leader board. Some players became so deeply involved in the competition that they submitted many thousands of photos apiece, most of them usefully contributing to a 3D model of that player's campus. To verify that our players' photos had more coverage and completeness than existing community photo collections, we compared our dataset to a Flickr set of Rome reconstructed by Rome in a Day, and to Flickr photo counts of each campus. The photos collected by PhotoCity are vastly more relevant to our purpose of 3D modeling than Flickr photos, especially of underrepresented locations like school campuses.

(a) Lewis Hall (UW)

(b) Sage Chapel (Cornell)

(c) Uris Library (Cornell)

Figure 4.9: Buildings reconstructed in detail during the PhotoCity competition.

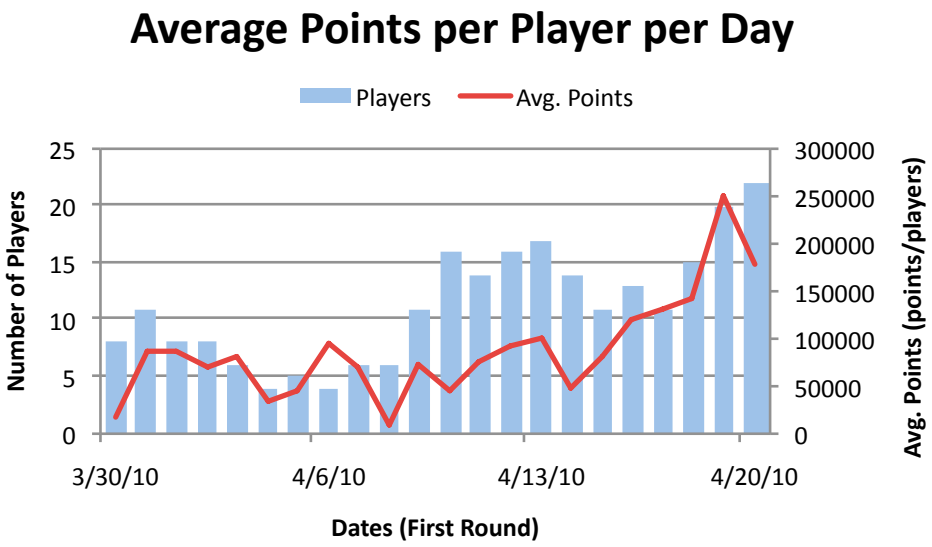


Figure 4.10: This graph shows the number of active players who submitted photos each day, and the average number of points per active player each day (total number of points scored that day divided by active players). Towards the end of the first competition round, players had learned how to take effective photos and earn more points per day: 100,000-250,000 points on average (and generating the same number of geometric 3D points).

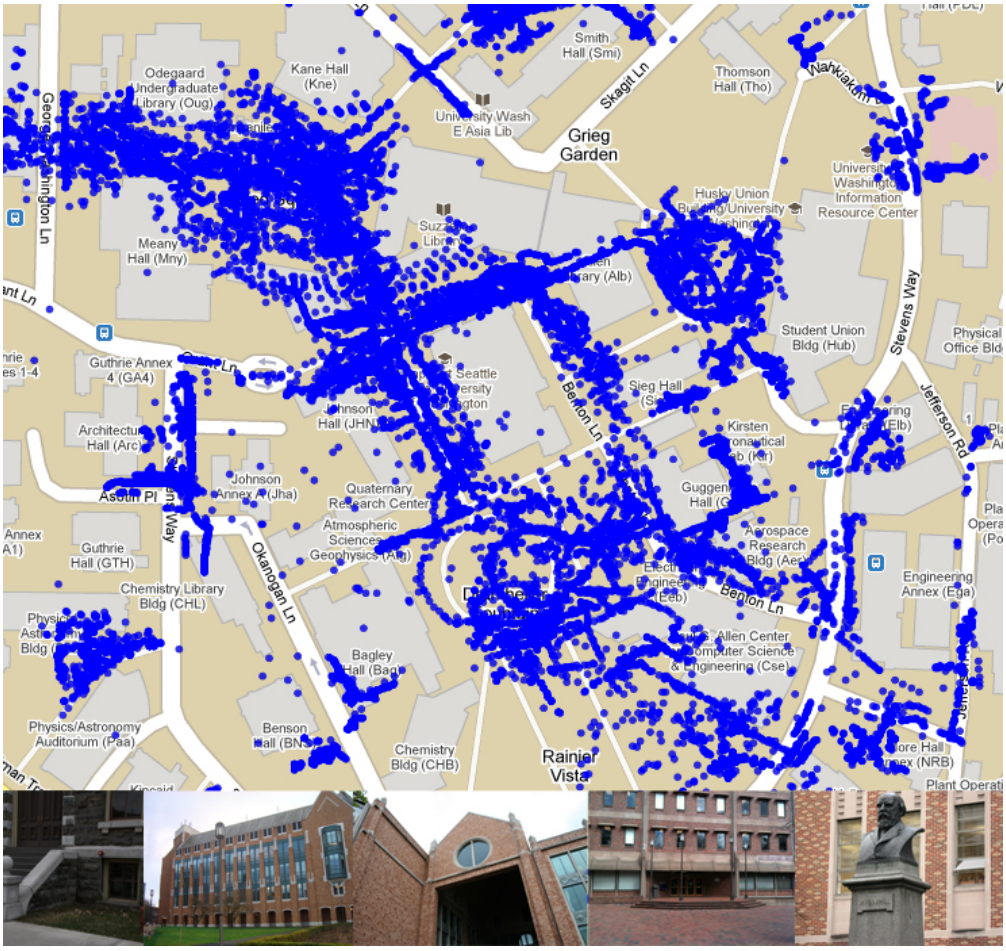


Figure 4.11: Locations of PhotoCity photos (where the photographer was standing) on a map of the University of Washington campus. Photos densely cover the parts of campus active in the game, especially walkways and open areas between buildings. Real photos from the game shown below the map.

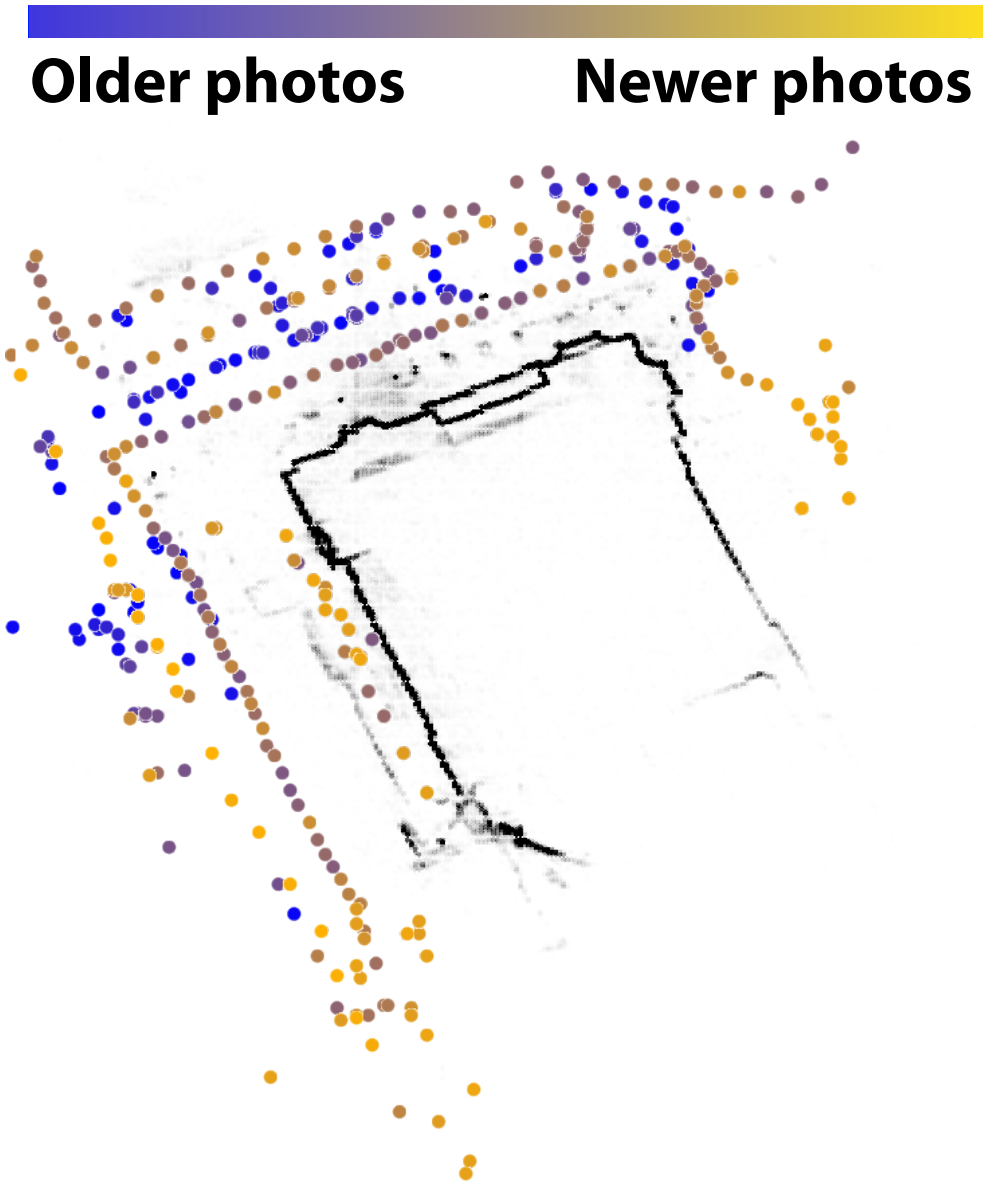


Figure 4.12: The walls of a building are shown in black and camera positions are shown as colored circles, colored in the order they were taken and submitted to the game, with blue being the oldest photos and yellow being the newest photos. The top-left (north-western) faces of the building were photographed first, then a very straight and deliberate path of photos was taken slightly closer to the building, and finally, photos were taken from completely new viewpoints (from the south and east) as the building grew.

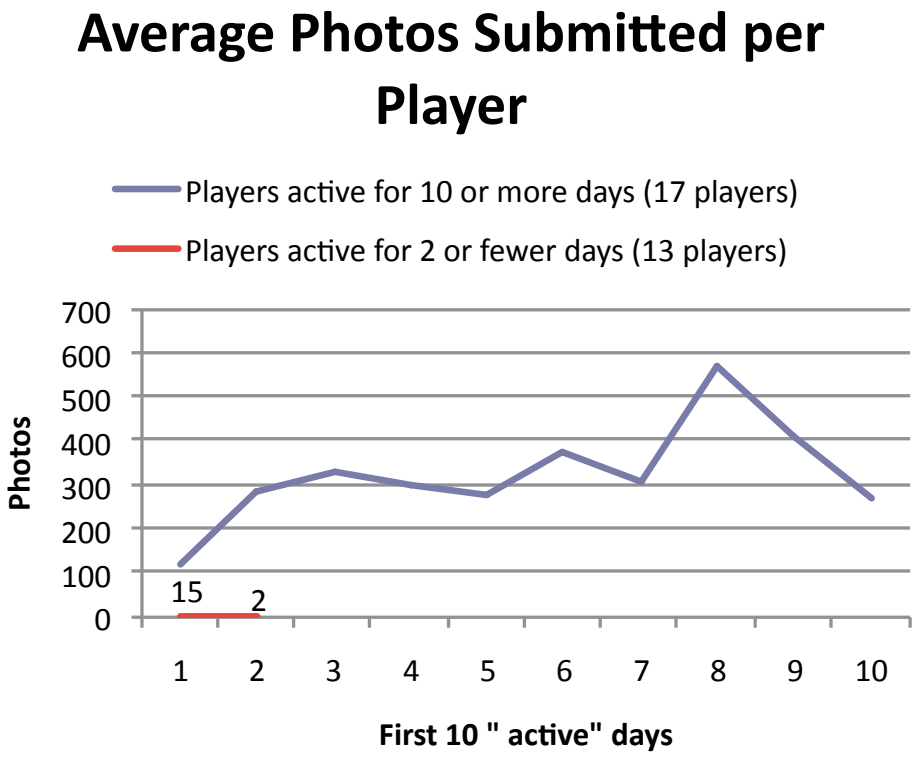


Figure 4.13: The most striking difference between players who became heavily involved in the game and played for many days (10 or more) was simply the number of photos each group took on their first few active days. Players who played very little (one or two days only) took very few photos their first day. Players who wound up playing a lot (10 or more active days) took almost 10 times as many photos their first day, and even increased the number of photos they took each day for the first several days, until leveling off.

Chapter 5

POINTCRAFT

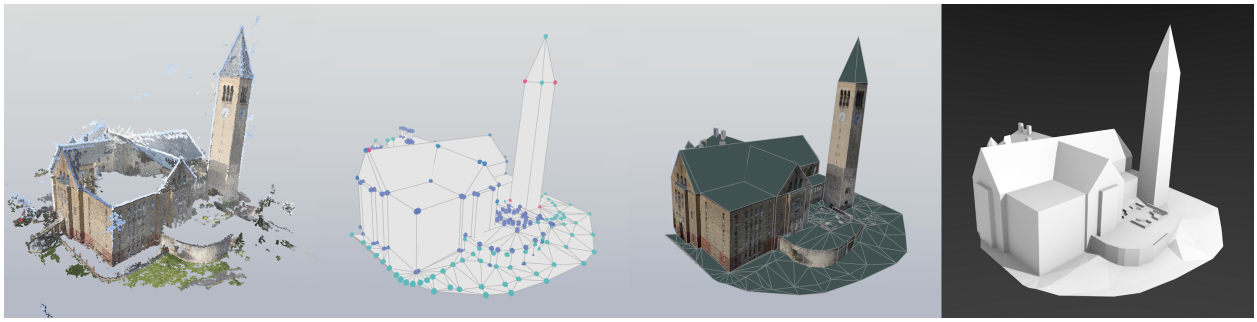


Figure 5.1: Point cloud, polygons and pellets, model with texture, and rendered geometry modeled in PointCraft.

The next CDCV system is PointCraft [193], a 3D modeling tool to complement the crowdsourced photography game PhotoCity. PhotoCity produces noisy, incomplete point clouds from thousands of player photographs, and automatic methods do not currently work to transform these point clouds into clean, concise, usable geometry. PointCraft is a tool to harness the power of existing players to model these point clouds with human interaction. Drawing on first-person navigation skills many game players have already developed, and specifically on the first-person modeling skills of Minecraft, PointCraft provides a new opportunity for players to contribute to the overarching goal of reconstructing a virtual model of the world. This chapter illustrates an in-depth example of the property of CDCV systems relating to interaction: PointCraft enables users to convey their higher level expertise about the shape of architectural structures by directly tracing point clouds and constructing geometry.

5.1 Introduction

I introduce PointCraft, a game-inspired tool for interactively tracing point clouds in 3D. PointCraft is designed for working on point clouds of buildings (left image in Figure 5.1, Figures 5.3, 5.4, 5.11) generated automatically from thousands of photographs (from different angles) captured by players of the game PhotoCity [198]. PhotoCity's overarching goal was to reconstruct 3D models of the whole world, but there was no way to get clean, concise, useful geometry from the noisy and incomplete point clouds produced. Some automatic methods like Poisson Reconstruction [79] make no assumptions about the fact that we are reconstructing urban architectural scenes (producing blobby artifacts as shown in Figure 5.2) while others [63, 119] are too constrained to work with the variety of models found in PhotoCity. The point clouds themselves are easily understandable by humans, and through PhotoCity, there is already a source of players who are interested and invested in the photography side of the problem. Our goal is to provide a new tool for existing PhotoCity players to complete the geometric modeling process, but make it accessible to new players familiar with traditional videogames as well, so that more people may creatively contribute to the modeling process.

To understand PointCraft's contribution, we first need to describe point clouds and where they come from. Point clouds are collections of 3D points and may be obtained from a variety of sources, such as LiDAR scanners, depth-sensing cameras (e.g., Microsoft Kinect, Leap Motion) or image-based techniques such as Structure from Motion, which was used by PhotoCity. Unlike lines or polygons, points are zero-dimensional geometric objects, and therefore hard to visualize. Depicting each point as a 3D sphere (for instance) causes nearby points to coalesce into a blobby mess. Distant points are often visible in between foreground points, causing visual confusion when the depth of the geometry is hard to discern.

A traditional 3D modeling tool such as Maya is too complex and cumbersome to expect an average PhotoCity player to use. Furthermore, point clouds pose their own challenges for interpreting, navigating, and manipulating. First, it can be difficult to interpret depth when

distant regions of the point cloud show through close regions. Also, it can be disorienting to navigate large models (e.g., reconstructions of urban scenes) using camera controls like the arcball, traditionally designed for viewing smaller objects. Finally, it can be frustrating to interact with and try to select points on a point cloud when there is no explicit surface to point a cursor at.

In order to develop a modeling environment that would seem familiar and easy to use, we took inspiration from the first-person voxel world-building game Minecraft, popular with millions of people around the world [148]. We found that providing users with natural walking (and flying) controls allowed them to navigate large point clouds in a more familiar way. Such motion also provided parallax, a depth cue otherwise missing from the point cloud, which helped the user understand the scene.

Similar to Minecraft's use of voxels as its modeling primitives, PointCraft uses a spherical *pellet* as its modeling primitive. The point cloud becomes not just a visual guide to the modeler, but a *virtually tangible* guide, something the modeler can grab on to and attach pellets to. As a result, modeling the 3D shape becuomes as easy as *tracing* the 3D point cloud, in the same way someone would trace a picture in 2D.

PointCraft not only makes the interactive tracing and modeling of point clouds possible, it addresses the difficulty of working with point clouds by combining the interaction and navigation into a single mode. This first-person modeling paradigm is an excellent fit for our domain. Our primary contribution is the PointCraft user interface itself and demonstration of its use to trace point clouds from PhotoCity. By building this new tool, we are closer to achieving the original goal of PhotoCity, to collaboratively and collectively reconstruct a 3D model of the world. We also wish to emphasize the idea of adapting familiar game controls to new, serious purposes where human intervention is incredibly valuable for solving problems that automatic techniques struggle with. Additionally, by extending an existing game with a purpose like PhotoCity with a new, related tool, we hope to widen the potential player audience and allow many different ways for players with different skills and interests to contribute to common goals.

5.2 Related Work

Games with a purpose leverage human common sense, intuition, and/or problem solving abilities to solve real world problems. Chapter 2 has more details. In some cases, the player simply contributes labels, as in von Ahn's ESP Game [207], but in other games, players use custom tools to control deep, domain-specific interactions. In the game Foldit, players manipulate virtual proteins in 3D in order to predict their folding structures [35], invent their own folding algorithms [33], and even design and restructure proteins [51]. Likewise, players of Eterna are discovering new rules for RNA structure design [115]. Our tool PointCraft has a similar premise to Foldit, in that there is a 3D problem that computers struggle with (automatically reconstructing noisy point clouds), but with the right tools, humans can use their own intuition and judgements to fill in the details and accomplish the task.

Before we talk about the ways humans can be involved in the task of modeling geometry from point clouds, we must discuss the history of automatically generating meshes from unstructured point clouds. Over the last two decades, automatic methods have successively improved [79, 2, 13, 3, 90], but they rely on data to be both accurate and reasonably complete without holes. Real world point clouds, especially those generated from photographs captured by multiple users with different cameras, usually violate both these assumptions. Using automatic methods on such point clouds results in reconstructions that look like Figure 5.2.

The type of model we desire is a simplified polygonal model, suitable for importing into Google Earth. To borrow a phrase from Chauve et al. [24], we would like "concise and idealized" geometry. Several automatic methods exist for generating piecewise-planar models from point clouds. Some use RANSAC [169] to generate candidates, while Chauve et al. use a region-growing approach. The approaches of Nan [138] and Li [119] incorporate a-priori knowledge about the structure of the input data (architectural buildings with repeated structure and small objects composed of basic primitives, respectively) which precludes their use on other types of point clouds. These automatic methods could work for portions of PhotoCity point clouds known to be highly structured and repetitive, but in general, the

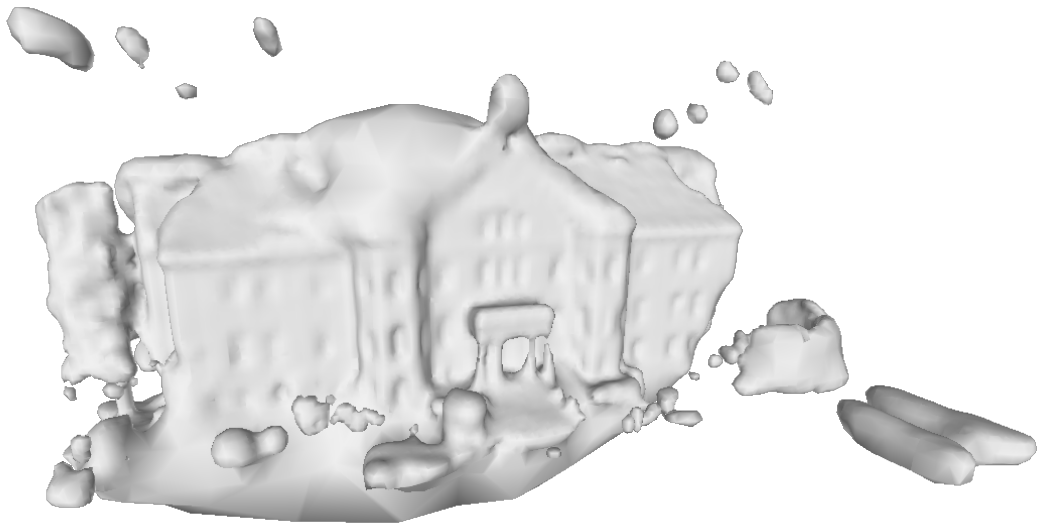


Figure 5.2: Automatic Poisson Reconstruction of Lewis Hall point cloud. Because the algorithm is unaware of architectural motifs and context for the scene, it smoothes out sharp edges, creates floating orbs out of noise and loose points, and makes unrealistic approximations where data is missing.

models we looked at were too complex and heterogeneous, containing many different types of architecture as well as plants and non-architectural objects.

Some systems exist which allow interactive modeling from point clouds, though they are too limited or too specific for our purposes. PointShop3D [227] was the first publicly-available software that allowed users to interact with the point cloud data directly. Pauly et al. [147] extended it, allowing the user to directly manipulate a hybrid surface model inside their software. Weyrich et al. [215] added tools for touching up the severe scanning artifacts present in many point clouds. Weyrich's tools in particular are designed for mostly complete point clouds and therefore not very effective when large swaths of geometry are absent. For instance, in Figure 5.1, the point cloud is missing its roof, which we were able to fill in using

PointCraft. The work of Colburn et al. [29] lets users place walls of a house using a point cloud as a visual guide. Since the end goal is *re*modeling houses, their system does not support the creation of arbitrary geometry, while PointCraft does.

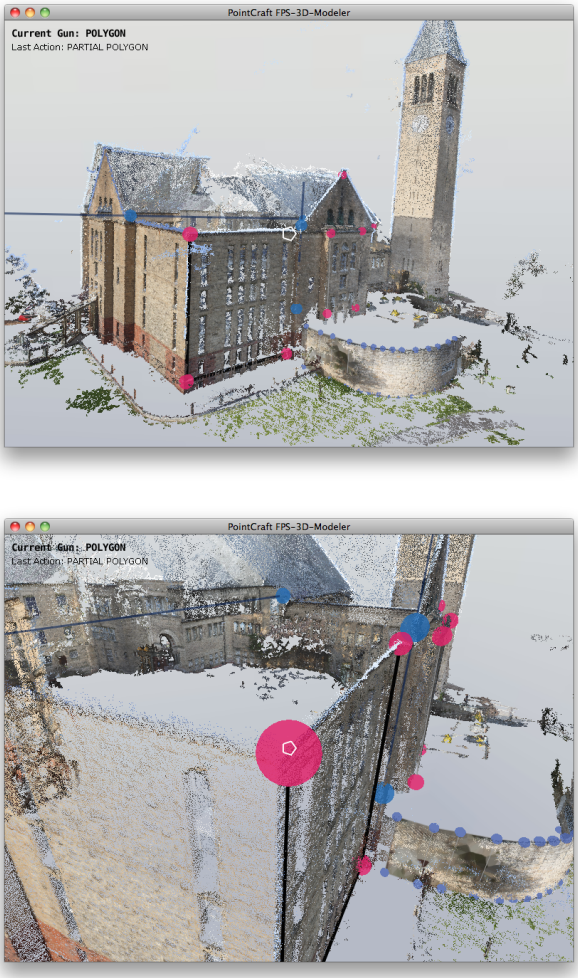


Figure 5.3: Two screenshots of the PointCraft user interface showing a user defining a polygon.

Since our point clouds are generated from images, we now discuss similar image-based modeling approaches. These techniques involve tracing on images or video frames and have proven to be quite powerful [43, 140, 175, 200] and capable of producing clean, yet detailed

models. Debevec et al. [43] exploited symmetry in architectural structures to recover geometry. More recently, Sinha et al. [175] showed how vanishing lines (pre-computed from the images) can be used as a visual guide for letting the user draw geometry on images. However, deciding which photos to present to the user from among potentially thousands of images is a non-trivial task. Further, using an image as a viewpoint precludes modeling from other vantage points (such as overhead views or extreme close ups).

There have been attempts to simplify 3D modeling via sketch-based interfaces [82, 142]. These systems leverage a more "natural" interface (typically a stylus on a touch-sensitive tablet) thereby enabling any user to create 3D geometry. However, by working on a 2D surface, the user does not have direct control over the created 3D geometry. Schmidt [168] added image-space linear 3D scaffolding to act as visual constraints, thereby giving the user more precise control over the 3D authoring process. PointCraft follows a similar approach, letting the user create 3D scaffolding to guide construction of primitives and/or more scaffolding.

Interactive 3D modeling software has traditionally been dominated by powerful, but hard-to-use high-end software such as Maya, SolidWorks, and CATIA that takes weeks to learn and months, if not years, to master. Our approach with PointCraft was to make the interface as simple and familiar (especially to people who have played Minecraft and other first-person games) as possible, allowing non-expert users to quickly become productive with it.

5.3 User Interface

PointCraft is heavily inspired by Minecraft, "a game about placing blocks to build anything you can imagine." In Minecraft, players can build anything they want, as long as they can represent it using blocks. With PointCraft, we wanted to give users the ability to easily model real-world scenes, provided they're available as point clouds.

In designing PointCraft, we borrowed the following ideas from Minecraft:

- First-person navigation (using the mouse to look around and keys to move forward, backward, left, right, up, and down) which in our case provides motion depth cues and

a natural 'walking' metaphor for navigating visually complex point clouds.

- Combined navigation and modeling, allowing the user to move around their work as they build, instead of explicitly switching modes.

- Tools based on a simple primitive (in our case, a spherical pellet instead of a Minecraft block) that are easy to explore and experiment with.

- A minimal heads-up display (Figure 5.3) showing only the current tool, the user's last action, and the tool palette. Users change tools with the number keys (a convention in shooter games), limiting the tools directly accessible at any point in time to just ten.

- An extended tool inventory from which users can customize their on-screen palette. (See Figure 5.6).

5.3.1 First-Person Navigation Controls

Our design goal with PointCraft is to allow videogame players to interactively trace point clouds, but navigating and interpreting point clouds can be confusing and disorienting. Multiple layers of 'surfaces' can be visible at the same time, especially when the point cloud is imperfect and incomplete. Additionally, cloud data without normals or with noisy normals lack cues about orientation generally provided through shading on solid surfaces. Luckily, motion (of the viewpoint) provides us with the depth and orientation cues otherwise missing from static images of point clouds.

Since 3D games have already trained players how to navigate large outdoor scenes via first-person controls, it is a natural design choice to exploit this in a game with a purpose that requires navigating 3D environments. Many of our point clouds are of buildings and large outdoor scenes featuring multiple buildings, and we have oriented them to give them a sensible up-direction. Users can walk forward and backward and strafe left and right with the conventional keys while changing the direction they are looking with the mouse. Walking

forward always moves the user horizontally in the viewing direction. This keeps the user attached to an invisible 'ground' plane and facilitates creating horizontally-aligned details.

Traditional modeling tools like Maya and SketchUp separate navigation from modeling, while many games allow players to interact with objects and shoot weapons while they move and look around. In most modeling tools, changing the viewing angle takes place in a different mode than creating or interacting with geometry, and the user must consciously switch between modes. When modeling is part of a first-person game such as Minecraft or the physics simulation sandbox Garry's Mod from Valve [57], navigation and object manipulation are the same mode. The player 'looks' at an object and clicks on it to interact with it. Due to the confusing visual nature of point clouds, combining interaction with first-person navigation allows depth cues from motion to help the user make sense of the point cloud. As users work, they can quickly check their modeling work from different angles as they go. If they make an error, they can seamlessly undo and try again without switching back and forth between navigation and modeling.

In the following section, we describe the first-person tools for building geometry and interacting with the point cloud.

5.3.2 Placing Pellets with the Pellet Gun

The most basic modeling operation in PointCraft is placing handles in the point cloud. We provide users with a shooting mechanism that lets them aim crosshairs in the center of the screen, and click to fire pellets at the nearest point in the point cloud, thereby specifying where to create these handles.

The *pellet gun* shoots spherical pellets that animate towards the point cloud, checking for intersections with nearby points and pellets, and stick to the point cloud at the first location they hit. When the user shoots, they see a spherical pellet (circle) emanating from their position in the direction they are currently looking. The pellet flies off toward the point cloud, eventually hitting and sticking to it. Since the pellet is represented by 3D geometry, perspective foreshortening causes it to shrink as it flies farther away, thereby conveying a

sense of distance. The pellets can also stick to other pellets and user-created scaffolding. If a new pellet hits an existing pellet, the pellets combine into a single spherical pellet.

The pellet also provides auditory feedback upon impact, letting the user know that the pellet has landed. The delay before this sound provides a secondary indicator of distance. Additionally, different noises indicate whether the pellet has snapped to a point in the point cloud or an element of the user-created scaffolding.

5.3.3 Modeling Tools

PointCraft provides a number of *construction* and *editing* tools that are quickly accessible through the number keys 1–9 (a built-in online sharing tool is mapped to number key 0). Construction tools generate pellets and/or primitives, whereas editing tools operate on existing pellets in the scene. Every construction tool fires pellets of a different color (shown in Figure 5.5). Firing a new pellet at an existing pellet combines the two, and the new pellet snaps to the old one's position.

5.3.4 Polygon Tool

The most basic tool is the polygon tool, which shoots pellets to form vertices and connects them by edges. By firing a sequence of four pellets that starts and ends with the same location, the user can form a triangle. Including more pellets in this sequence results in the construction of a triangle fan around the first point, allowing the user to construct arbitrary planar polygons without switching tools.

5.3.5 Scaffolding Tools

Given a point cloud with no missing data, a user could construct a mesh using only the polygon tool. However, the resulting shape might be sloppy, like tracing a picture without a ruler. Thus we introduce scaffolding tools, which not only make the reconstruction neater, but also allow the user to extend the model past the edge of the point cloud. The simplest

piece of scaffolding is a **single pellet**, which can be used to define polygons or other scaffolding. There are also **lines** (defined by two points) and **planes** (defined by three points). Once scaffolding has been defined, the user can fire pellets at a line or plane in addition to the point cloud. In Figure 5.4, the user has used line scaffolds to create even, regular dormer geometry and defined the shape of the roof despite the point cloud containing no roof data.

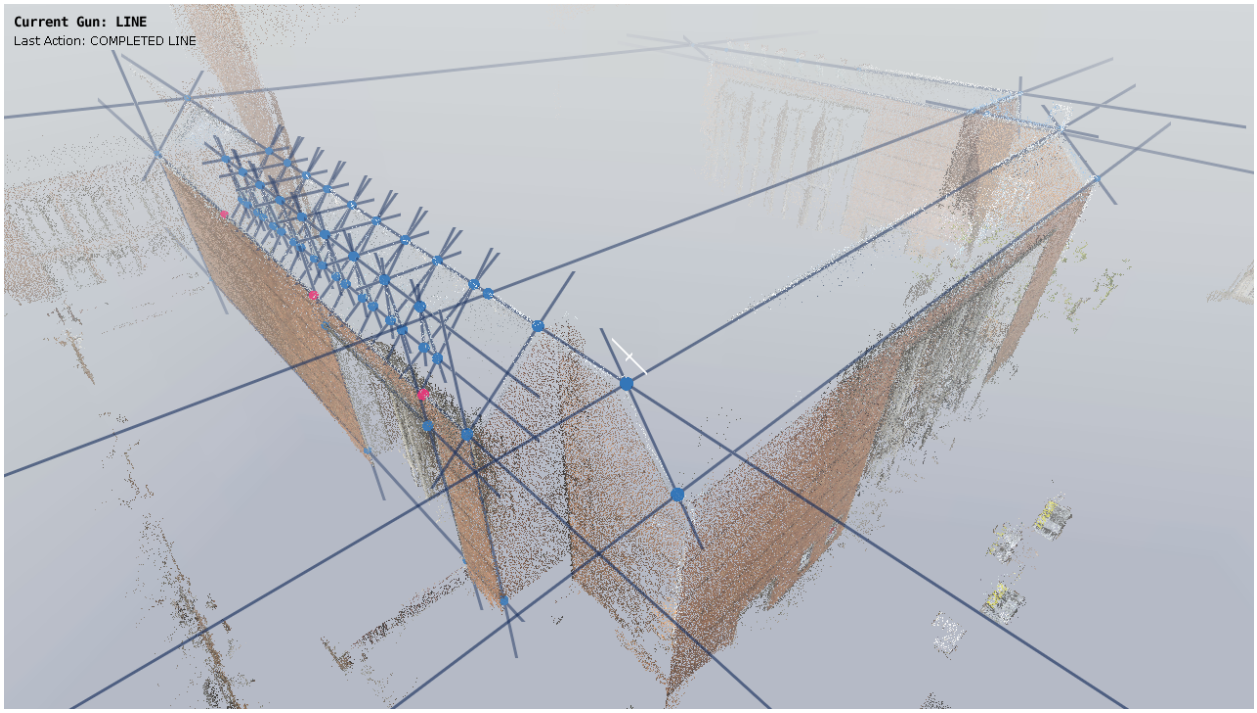
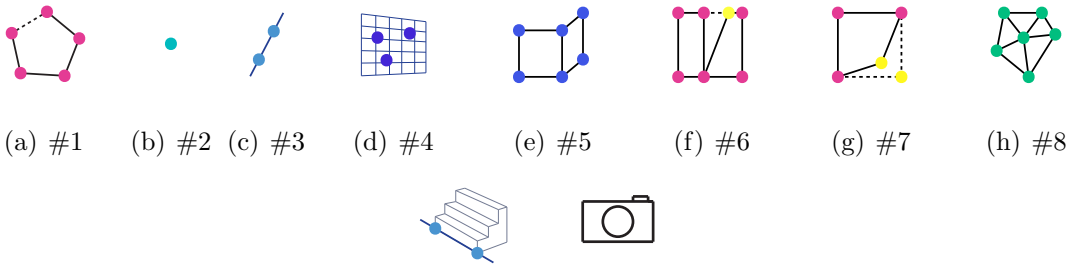


Figure 5.4: Line scaffolding for roof, walls, and repeated dormer structures.

5.3.6 Smart Tools

PointCraft's simplicity and versatility come at the price of speed. A user can create any mesh, but to do so using the above tools, the user has to place and connect every pellet. In order to minimize tedious, low-level activities, we offer some smart high-level tools to accomplish complex tasks.

In an effort to make reconstructing a building as easy as marking out its floorplan, we



(a) #1 (b) #2 (c) #3 (d) #4 (e) #5 (f) #6 (g) #7 (h) #8

(i) #9 (j) #0

Figure 5.5: These are the nine PointCraft tools accessible through the number keys 1–9. From left to right, they are: polygon, pellet scaffold, line scaffold, plane scaffold, wall, combine pellets to edit, drag to edit, triangulated mesh, and change direction of wall tool. The 0 key is a tool that lets users select a viewpoint and automatically share a screenshot and their in-progress geometry.

created the **wall tool**. The wall tool provides a shortcut whereby when the pellet hits the point cloud, it spawns two new pellets traveling in opposite vertical directions that stop when they find a gap in the point cloud, creating a line that spans the height of the point cloud at that location. The next time the wall tool is fired, it creates a new line attached to the pellet's impact location that is parallel to the first line and connected into a rectangle. Essentially, a user can create even wall segments with very few clicks. Figure 5.7 shows walls constructed with this method.

Since point clouds are not always perfectly vertically aligned, PointCraft allows the user to specify what 'vertical' means, in the context of the wall tool. By shooting a 'direction picker' pellet at any line, the user can choose that line's direction for the wall tool, thus turning the wall tool into a generic parallel line tool to quickly make stairs or slanted walls.

For shapes that are more curved and organic, we created a **meshing** tool, which computes the 2D triangulation of the 3D points. We used this to manually create coarse meshes of the bushes surrounding the building in Figure 5.9.

Once we were committed to the Minecraft-style first-person modeling paradigm, there were many more highly specialized tools we could imagine, such as tools specifically for stairs

or windows, or tools that let users create regular circles, spheres, and boxes. Some of these regular-primitive tools we built and made available in an inventory inspired by Minecraft's tool and material inventory. From the tool inventory, users can choose which tools they want in their on-screen palette mapped to specific number keys.

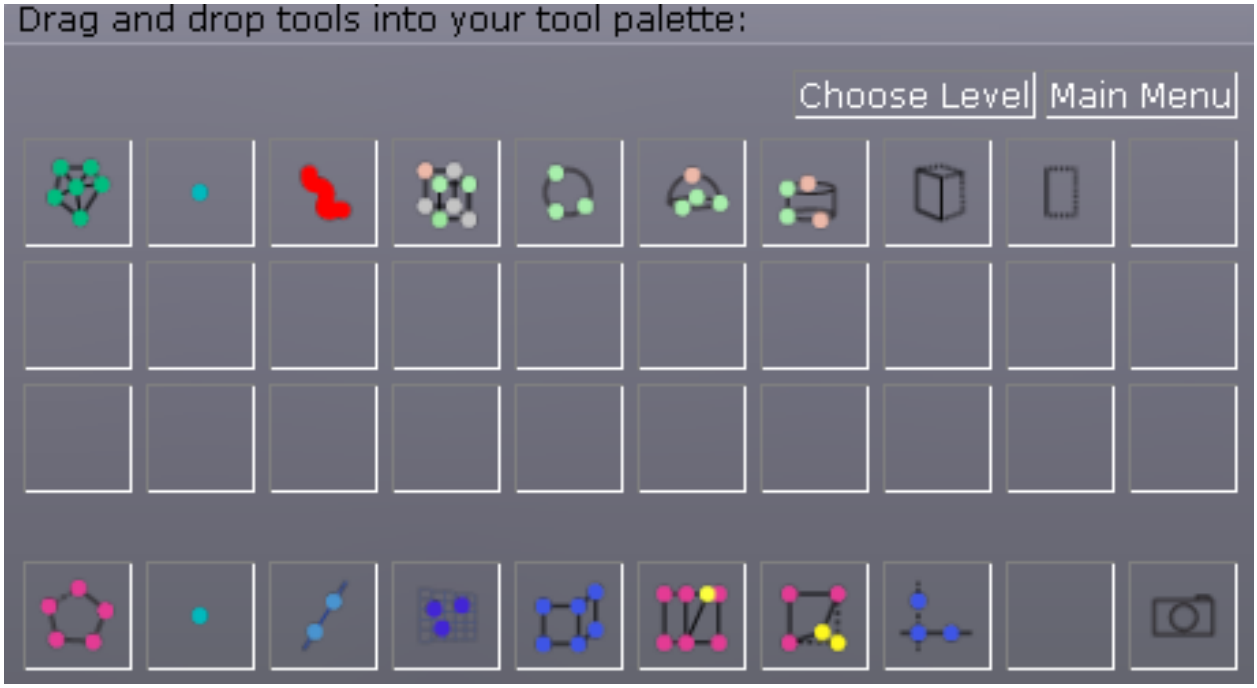


Figure 5.6: PointCraft users are not limited to nine tools; an inventory of additional tools lets them choose which tools they want to have immediately available through specific number keys. The additional tools include a paintbrush tool for selecting points, tools that make regular boxes, circles, domes, and cylinders, and tools that let the user extrude lines and polygons.

5.3.7 Editing Tools and Undo

We provide a small set of editing tools that are powerful, yet intuitive: dragging to edit the positions of the pellets and snapping them to other pellets.

We also provide unlimited undo via 'Ctrl-Z', which is extremely useful for quickly correcting errant pellets and polygons. The pellet gun may not position the pellet where the

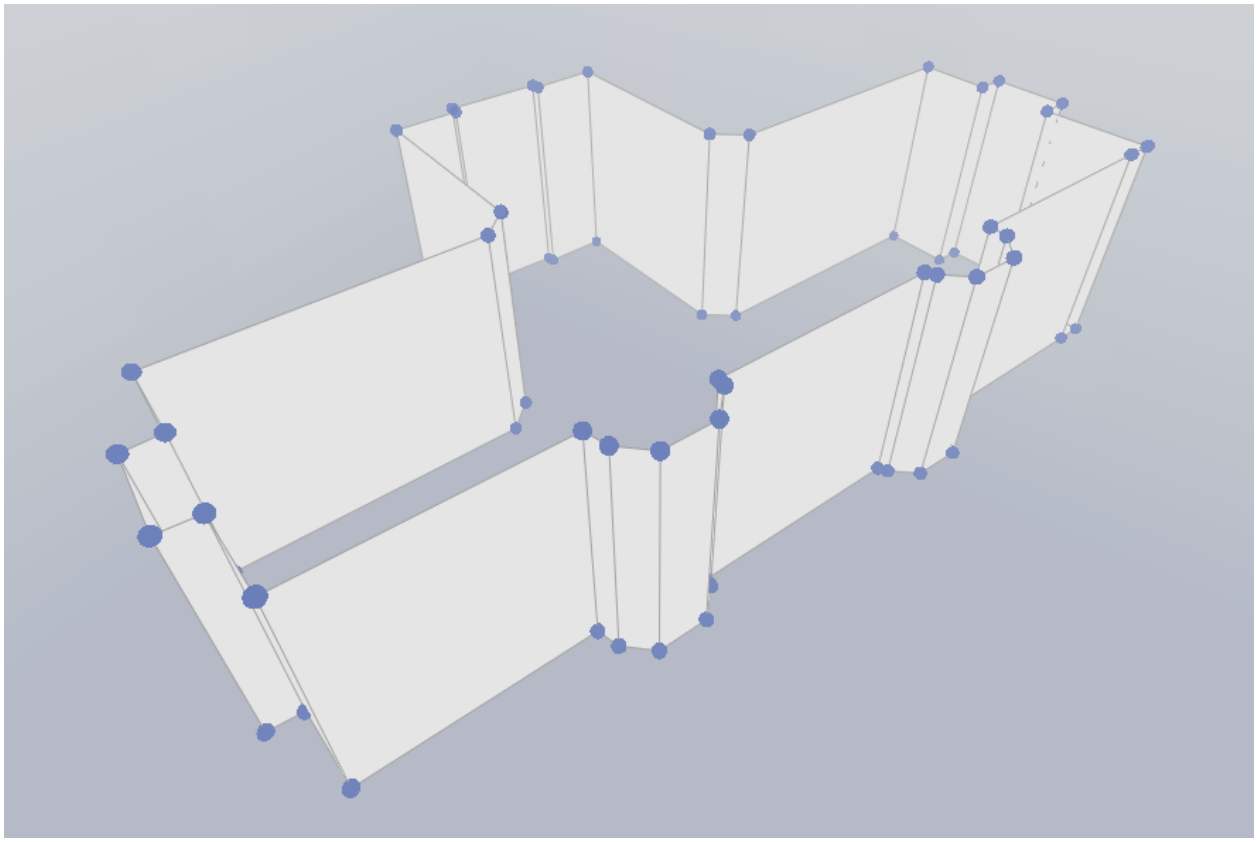


Figure 5.7: The wall tool makes constructing straight walls as easy as tracing a floorplan.

user intended, especially if there were unnoticed, closer point cloud points that the pellet stuck to instead of its intended target. But the integrated navigation and easy undo allows the user to quickly check the pellet's placement and easily try again from a different vantage point if the first attempt did not succeed.

5.3.8 Additional Controls

Users can toggle different display properties through several additional keys, such as hiding or showing the point cloud, pellets, or scaffolding, and changing the size of the points or the amount of fog. Adjusting the fog and point size helps when viewing the point cloud up close by providing additional depth cues. Because not all point clouds have the same size or point

density, users can adjust the speed at which they move around the model, and the size of the pellets' collision radius.

5.3.9 Sharing and Collaboration

Built into PointCraft is the option to automatically share the model along with a screenshot. The user enters the camera tool mode, navigates to a nice viewpoint and clicks, whereby PointCraft uploads a file containing the user's geometry, along with a screenshot from that viewpoint. The screenshot is shown on the PointCraft website for others to download. Other users, or even the same user, can load existing geometry along with the relevant point cloud and continue modeling where the original user left off. Figure 5.10 shows a model built by multiple people working on different parts of the same point cloud. In the current iteration, PointCraft does not support multiple simultaneous users, although it could, if we designed suitable avatars for players to be aware of each other's actions. We would likely borrow Minecraft's model in which players see each other as characters in the world, and can interfere with each other's work, but where social etiquette mostly keeps players collaborating. In the future, if there were a source of urban-scale point clouds that needed to be converted into clean, low-polygon geometry, we believe PointCraft users could collaboratively tackle this task.

5.4 Results

Because PhotoCity was offline at the time PointCraft was developed, we used models from the PhotoCity Archives [191], including the buildings of Uris Library (at Cornell University), Lewis Hall, and Red Square (both at the University of Washington). These point clouds were automatically generated from a few thousand player-captured photographs and contain hundreds of thousands of points each. Automatic reconstruction methods completely failed on these point clouds, generating surfaces like the ones shown in Figure 5.2. It would take expert-level knowledge and effort to repair these meshes with traditional 3D modeling tools. With PointCraft, users are able to trace the point clouds directly and even fill in appropriate

geometry where there was no point data. The resulting geometric models are clean and concise.

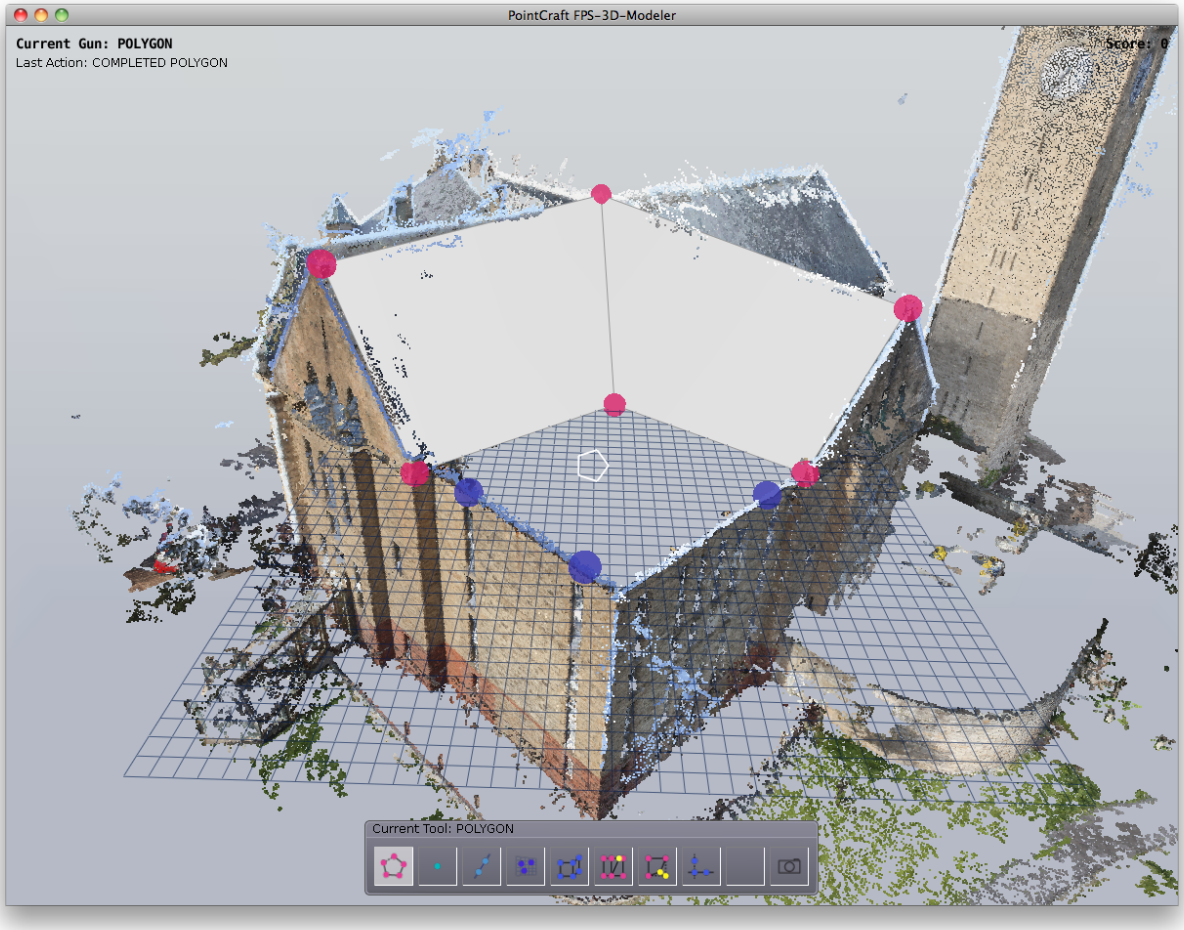


Figure 5.8: A plane scaffold being used to construct the roof where the point cloud contains no data. See Figure 5.1 for a view of the point cloud with no roof.

5.4.1 Lewis Hall and Uris Library (Cornell)

We now show author reconstructions of Lewis Hall and Uris Library, two models that were used as case studies when designing and developing PointCraft tools.

Because nearly all the images are taken from ground level, the point clouds do not include

the roofs of the buildings. The model of Uris Library (Figure 5.1) demonstrates this lack of roof detail nicely. To model the roof in PointCraft, we constructed a planar scaffold and were able to place a pellet at a point on the roof that was invisible to all photos (see the pellet in the center of Figure 5.8). In total, it took an experienced user 65 minutes to construct the finished Uris model.

The model of Lewis Hall had a simpler roof and simpler wall architecture, but had a lot of detail around the front entrance of the building that we wished to reconstruct. We modeled the walls (Figure 5.7) and roof of Lewis Hall in under 10 minutes, and spent another two hours on the entryway, including the pillars, stairs, stair railings, and foliage. Figure 5.9 shows the final result.

5.5 Pilot Study

In designing a modeling tool for tracing point clouds, we borrowed navigation and modeling paradigms from Minecraft and adapted them to point clouds. To make sure we had not created another 'experts-only' modeling tool, we ran a small pilot study and gave PointCraft to several novice users who had never used PointCraft before, had no experience with point clouds, and limited or zero experience with traditional 3D modeling tools. These users, four males in their twenties, did have experience with Minecraft and with gaming in general. We acknowledge that this is a small, biased sample and that there are many interesting questions remaining, such as skill transfer and how the game-inspired interface of PointCraft changes the overall experience of participating in such a game with a purpose.

During our pilot study, we sat down with four users and asked them to model a point cloud from PhotoCity of buildings in a location called Red Square on the University of Washington campus. We spent five minutes at the beginning going through each of the tools and letting each user scope out the scene. Following this, each user was allowed to choose what to work on, with no time limit given. They enjoyed the familiar FPS-style navigation and spent upwards of 45 minutes absorbed in their own modeling work. One of them, who had previously attempted to use Maya on his own but was unable to make any meaningful

progress, praised the usability of PointCraft.

While each user worked individually, they modeled separate portions of the same multi-building point cloud. The collaboratively-built model combining their work is shown in Figure 5.10 with each user's work in a different color.

Even among these four users, we observed PointCraft's small set of tools being used in different ways. One user set up scaffolding first and then spent little time editing (Figure 5.4 is the work of this user), while other users created freehand polygons and relied on editing for cleanup. This demonstrates an ability to accommodate more than one working style.

5.6 Limitations of PointCraft

PointCraft struggles with fine detail because it is designed for creating large, flat polygons. When a point cloud is a dense, clean approximation of a surface, automatic methods such as Poisson Reconstruction [79] work well, but many point clouds that people wish to use with Poisson are not complete. PointCraft's strength is laying out coarse geometry to fill in missing regions, so perhaps it can be used to aid the automatic Poisson method.

We used PointCraft to complete a model of the Sistine Chapel reconstructed from tourist photographs. Naturally, most people take photos of the ceiling, and the presence of other people obscures the view of the ground, so the ground is not included in the automatic reconstruction. Using PointCraft, we were able to model planes to extend the walls and complete the floor. Figure 5.11 shows the new points sampled from PointCraft geometry in white and Figure 5.12 shows a view from inside the Sistine Chapel model after running Poisson reconstruction with the floor now included.

5.7 CDCV Properties and Outcomes in PointCraft

The CDCV property that this project explores most deeply is that of interaction. Like the direct manipulation afforded users in the protein folding game Foldit [35], PointCraft provide a user interface that lets users directly create and manipulate the geometry they think should exist, using the point cloud as scaffolding. The core computer vision system is

that of point cloud reconstruction, but only simple instances of that technology (e.g., fitting a plane to a set of points) are active in the system. The mappable data space of point clouds and possible 3D geometries is more abstract than the geographic map of PhotoCity. The final CDCV property of the system — providing feedback to users about their contributions — is the least strong in PointCraft. The textures that the system generates for the users' polygons are a form of feedback about how consistent that geometry is with what the photos captured about the real world scene. But most of the evaluation of the quality of the users' geometry is done by the users themselves and whether they successfully created what they intended to create.

As for CDCV outcomes, PointCraft demonstrates certain outcomes more strongly than others. Due to the scale of our study, the outcome of accumulating data in areas of user interest took the form of users choosing which of a small set of available point clouds they wanted to model, and which region within that model they wanted to focus on. This was not at the scale of PhotoCity seeds growing in five out of the seven continents but it still represented a path for user interest to direct how users contributed. However, PointCraft certainly represented a new way of building datasets of noisy architectural point clouds essentially annotated by humans with clean, mesh geometry. This level of annotation for point clouds is at the same level as human annotations on images (though boxes, scribbles, or polygonal outlines) that drives image understanding research. PointCraft also deeply embodied the outcome of engaging the creativity, skill, and diversity of the crowd, through the skills that players quickly learned in order to use the tools, and their creative and unique strategies for modeling different regions of different point clouds. Again, due to the scale of our study, the final outcome of motivating and focusing crowd effort was not seen as strongly as in PhotoCity. Ideally, PointCraft and PhotoCity would have been deployed at the same time in a synergistic ecosystem and output from one would have driven contribution opportunities in the other.

A dimension of PointCraft that is not part of the CDCV properties but still contributed to the utility and collaborative creativity of the system was the ability for users to easily share

their work, including a snapshot of their creation and the data for the underlying geometry. This provided an avenue for users to browse the work of others, get inspiration for what was possible, and to be able to directly download and build on top of the geometry started by another user. This supports a collaboration pattern of deep chains of contributions.

5.8 Conclusion and Future Work

Driven by the need to generate concise meshes from PhotoCity's point clouds, we have developed PointCraft, an interface for interactively tracing point clouds in 3D. Following our aim to retarget videogame players' existing skills, we have borrowed ideas from the first-person voxel world-building game Minecraft, designing a game-like UI that is familiar to such users. We have thus not only enabled existing players who are invested in PhotoCity to contribute, but also opened the door for new players to contribute to this part of the overall goal of reconstructing the world in 3D. PointCraft was successfully used to model PhotoCity point clouds, closing the gap between crowdsourced photographs and clean, concise 3D geometry.

By designing a game-inspired modeling interface with first-person navigation and shooting, we were able to harness players skills with previous videogames to a new, serious purpose. Furthermore, the first-person modeling paradigm seen in Minecraft worked exceptionally well with point clouds, which are otherwise difficult to navigate and interact with.

In the future, we hope to integrate PointCraft into the PhotoCity game. Our long term goal is to unify the two systems and see effort, ideas, and creativity flow back and forth between photographers and modelers. By creating more ways for players with different skills and interests to participate, games with a purpose will be able to attract a wide audience where many players with different interests and skills can collaborate on the same goals.

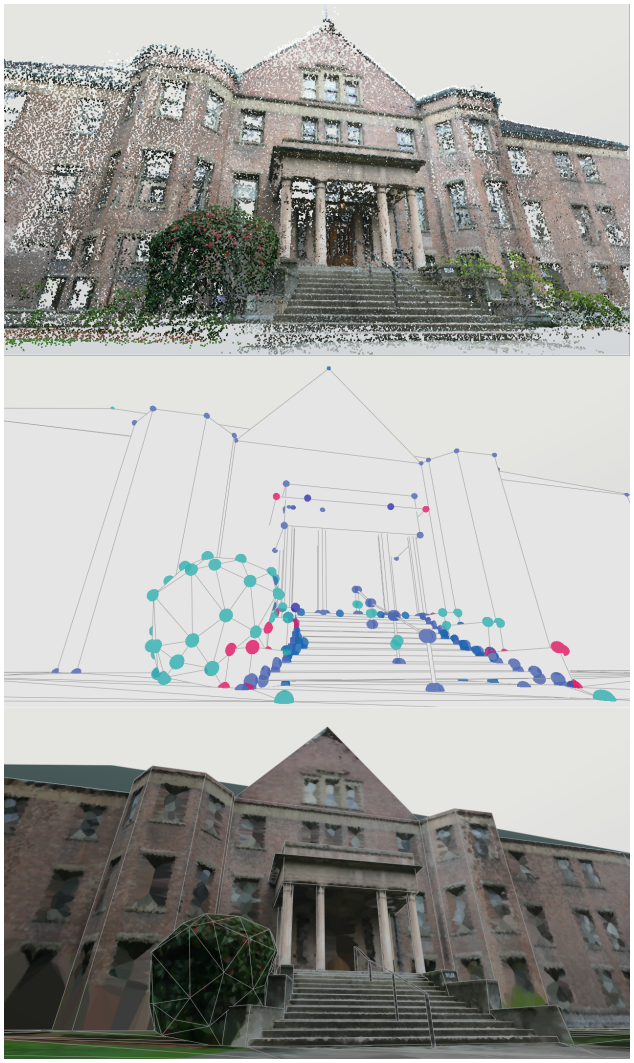


Figure 5.9: Lewis Hall model with pillars, stairs, handrail, and bush. The top image shows the point cloud. The middle image shows the PointCraft geometry and pellets. The bottom image shows the textured geometry.

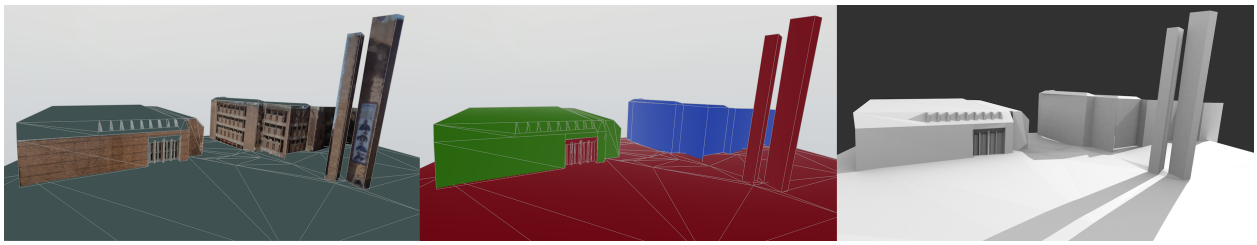


Figure 5.10: Multi-building model collaboratively built by multiple users.

Figure 5.11: PointCraft was used to add coarse geometry and sample new points along the walls and base of the incomplete Sistine Chapel point cloud.

Figure 5.12: The result of running automatic Poisson Surface Reconstruction on a point cloud of the Sistine Chapel augmented with points sampled from wall and floor geometry added using PointCraft.

Chapter 6

THE MEME QUIZ

This chapter describes the final CDCV system of this dissertation, a game with a purpose called *The Meme Quiz* [194] in which a human player mimics popular Internet memes, and the system guesses which expression the player imitated. The purpose of the game is to collect a useful dataset of in-the-wild facial expressions. The game was deployed with 198 players contributing 2,860 labeled images. In contrast to many data-gathering games that use interaction between humans to define the mechanics and verify the data, our game has an online machine learning system at its core that verifies data as it comes in and presents the results to players as feedback. As more people play and make faces, *The Meme Quiz* accumulates more data and makes better guesses over time. The focus of this chapter is on using an interactive computer vision system in the core of a CDCV data collection system to enable rich feedback about user contributions and the continued improvement of the system over time.

6.1 Introduction

Facial expression recognition is an important part of affective computing. Typically, only the six basic expressions of joy, sadness, fear, anger, surprise, and disgust are used in affective computing – a small set of facial expressions by all accounts. We are interested in extending the capabilities of automated expression recognition and in collecting a new dataset of facial expressions that includes many new expressions. To do so, we take advantage of the broad set of facial expressions that appear in Internet memes.

Reaction images [6], known by the shorthand *MRW* ("My Reaction When"), are a type of meme that portray an emotion in response to something said or experienced. "Not Bad

(a) Not Bad Obama (b) Not Impressed McKayla

Figure 6.1: Example Internet Memes portraying several different emotions. Do these emotions have obvious names, or is the picture itself a more concise way of conveying the emotion?

Obama" and "Not Impressed McKayla", shown in Figure 6.1, are two recognizable media images that have been elevated to meme status. *MRW* memes can also include non-human faces, such as "Grumpy Cat" in Figure 6.6(c). These reaction images may have value beyond entertainment; Figure 6.2 shows a *MRW* meme known as "Success Kid" annotated with a story about a user's breakthrough using reaction memes to convey emotional state to a therapist. Communicative expression-related memes would be useful to affective computing, where a primary goal is to assist people who struggle with reading or communicating emotions in their everyday lives.

Although reaction memes themselves are popular on the Internet, there is no data source of everyday people portraying these same facial expressions. Anticipating that imitating memes would not only generate useful data but also be amusing and compelling, we set out to build a game to crowdsource photos of people imitating meme facial expressions. Since our end goal is to use the collected dataset to train an automated system to recognize these new expressions, we decided to build the training and teaching of this system into the game.

In this paper we present *The Meme Quiz*, a game we developed where the player is asked to act out a meme expression and the system guesses which meme the player is imitating. Over time as the game collects more data, the system improves and is able to guess expressions correctly. We designed our game such that it does not require the expression

Figure 6.2: A "My reaction when" meme depicting a story about using MRW memes to convey emotional state to a therapist.

recognition technology to work perfectly; the fun of the game comes from the fact that the system occasionally makes mistakes. In fact, our system can start learning immediately and does not need to be bootstrapped with initial data. In the middle of deployment, we were able to adapt the game by adding new memes to impersonate, and we were able to monitor the health of the data over time to make sure the system was in fact learning these novel facial expressions.

Because our game uses online machine learning as its core mechanic, it is different from other crowdsourced data generation games. In the rest of this paper, we explore the space of crowdsourced data-generation games along two dimensions of *human agency* and *machine*

involvement, and describe how *The Meme Quiz* fits as a game with high agency for both the human and the computer. Our contributions are 1) the design and deployment of a game for collecting diverse facial expression data and 2) an exploration of the design space of data-gathering games along two axes: human agency and machine involvement, including advantages of building a game around an interactive domain-specific technical system.

6.2 Related Work

This section focuses on background work related to facial expressions and crowdsourcing of these expressions. Games with a purpose are highly relevant, and we discuss many games in Chapter 2, as well as later in this chapter in Section 6.3 on our proposed design space for data-gathering games.

Name	Subjects	Photos per Subject	Expressions
CK+	127	4 videos	6
Multi PIE	337	2,225 photos	6
MMI	90	20 videos+photos	6+AUs
AM-FED	242	1 videos	2

Table 6.1: Comparison of facial expression datasets

There are a number of existing facial expression datasets, such as CK+ [124], CMU Multi-PIE [73], and MMI [143], which have been laboriously captured and annotated with emotion and Action Unit (AU) labels from the Facial Action Coding System (FACS). These datasets have fueled facial expression recognition research for over a decade and Table 6.1 shows a comparison of these datasets.

These standard datasets are often collected in controlled lab environments with, at most, a few hundred subjects. In practical applications, face trackers must work on a wide variety of facial appearances and in many different lighting conditions, and on more than six expressions. Bigger datasets are necessary, as well as datasets captured in the real world in

realistic situations, such using a webcam or a front-facing camera on a mobile phone. Our game captures faces in realistic capture conditions, and includes many more expressions.

The AM-FED [133] dataset was also captured "in the wild" by recording subjects' faces as they watched Super Bowl advertisements on their own computers. As an incentive for allowing their reactions to be recorded, subjects were shown a chart of their smile activity compared to others, which is an interesting integration of computer vision back into the user experience. The expressions captured in AM-FED are spontaneous (or as spontaneous as they can be when the subjects are aware they are being recorded), but the videos were chosen to elicit joy only, so the dataset does not span a wide range of emotions, or even very extreme emotions. While our own dataset is posed, it includes the basic expressions as well as many more, captured in real world environments.

Capturing spontaneous expressions is difficult, as it requires subjecting users to unpleasant stimuli designed to elicit emotions such as disgust, fear, or pain, and different people might not be sensitive to the same stimuli. Recently, Li et. al. [118] and Yan et. al. [219] have compiled datasets of spontaneous micro-expressions using videos chosen to elicit emotions including joy, surprise, and disgust and encouraging subjects to try to hide their emotions. Zhang et. al. [225] have also captured a 3D dataset of spontaneous expressions by engaging lab subjects in different activities, such as playing an embarrassing game or experiencing an unpleasant smell. We believe acting out expressions is more fun for the participant than being subjected to unpleasant stimuli.

New datasets of labeled examples of facial expressions that span a wide variety of people, capture conditions, and emotions are critical to the future of automated expression recognition and affective computing. Especially compared to bringing subjects to act out expressions in-person, online crowdsourcing has the potential to recruit many more subjects and collect far more data. *The Meme Quiz* is one of many possible ways to realize mechanics and incentives of crowdsourcing facial data.

6.3 Design Space of Data-Gathering Games and Systems

Before we describe our game, we want to define the design space of games with a purpose (GWAPs), specifically those used for gathering data, and position *The Meme Quiz* within that space.

As mentioned in Chapter 2, Games with a purpose, such as the *ESP Game* [207], were first introduced to produce large, labeled datasets to be used as training data for computer vision and machine learning tasks. Since then, games including *BeFaced* [190] and *Motion Chain* [182] have been developed to generate new data. We will call these games *data-gathering* games, with *data-generation* games as a subset. Paid crowdsourcing through micro-task platforms like Amazon Mechanical Turk (AMT) are also a common way to gather and generate data. Not all games with a purpose are data-gathering games; some like *Foldit* [35], *Phylo* [89], and *EteRNA* [115] are about solving puzzles and understanding the human process of finding optimal solutions, rather than simply collecting a dataset. *The Meme Quiz* is ultimately about collecting a dataset, but also understanding the system's learning process.

In order to understand what makes our data-generation game *The Meme Quiz* different from other data-gathering games, we examine a design space split along two different axes: **human agency** and **machine involvement**. Figure 6.3 shows the game design space split into the axes of the human agency and machine involvement, with the games discussed in this section.

6.3.1 Human Agency

How much choice does the player have? Do players get to be creative and try different options, or is the game expecting one objective right answer from them? In her book *Hamlet on the Holodeck* [137] Murray says of agency, "When the behavior of the computer is coherent and the results of participation are clear and well motivated, the interactor experiences the pleasure of agency, of making something happen in a dynamically responsive world." Games

are about giving players choices, especially choices that have a clear and purposeful impact on the game world, so the games mentioned in Figure 6.3 all have at least some human agency. Agency can blend into creativity and artistic self-expression, with drawing games like *Picard* [195] and with Zitnick's Abstract Scene clipart-based creator[226] discussed below.

Standard non-game labeling tasks on AMT, such as tasks posted by the requester "Tagasaruis" have low human agency, with propmpts such as, "Determine the number of people in an image or whether the scene is indoors or outdoors." The human does have to examine the image and provide an answer, but there is a single, objective answer that is expected.

In von Ahn and Dabbish's *ESP Game* [207], a game in which one player tries to guess what another player looking at the same image would provide as a tag, there is slightly more human agency. While the tags should ultimately capture what is important in the image, the player gets to make that choice, and their success in the game is influenced by the choices of other players. The *ESP Game* is more open to gathering subjective labels from players than many labeling tasks on AMT.

In *The Meme Quiz*, players have even more agency and responsibility. They must choose faces to imitate and then control their own faces to immitate those expression. They can even change their hair or put on makeup to better match a face, as some of our players have done. In addition to simply providing a label, our players also provide the unique details of their own faces.

6.3.2 Machine Involvement

The game *BeFaced* [190], a casual tile-matching tablet game, is similar to ours in that its purpose is also to collect expressions. Unlike *The Meme Quiz*, *BeFaced* only focuses on the six basic expressions. We consider *BeFaced* to have medium machine involvement, since a live face and expression tracker and runs as part of the game, but is secondary to the tile-matching mechanic. The off-the-shelf expression tracker used by *BeFaced* does not improve over time with more data. In fact, instead of the machine adapting, the players appear to have adapted to the deficiencies in the tracker, avoiding challenging faces and tilting the

tablet to make the recognition work [190].

The *ESP Game* [207] on the other hand is primarily concerned with obtaining appropriate labels for images, so there is no system built into the game that actually tries to use the labels. The only machine intervention is identifying taboo words, words that have already been used to label an image, which force players to come up with additional labels.

In contrast to these examples, *The Meme Quiz* lies at the high end of machine involvement. We know that face and expression tracking do not always work, because the space of faces, expressions, lighting, and pose is much more vast than what these systems have been trained on. We chose to build a full face tracking and expression recognition system into our game, and incorporate that unreliability into the game mechanics, making *The Meme Quiz* about experiencing and improving the limitations of facial expression recognition systems. The human player has a large role in acting out expressions, but the machine is heavily involved as well, as the main mechanics of the game depend on the underlying system learning and improving over time.

6.3.3 Additional Examples

PhotoCity (described in Chapter 4), like *The Meme Quiz*, is a game built around a computer vision pipeline. In the case of *PhotoCity*, the underlying system automatically generates 3D models from player photographs, and computes how much 3D geometry each new photo contributes. The 3D reconstruction system underlying *PhotoCity* is more complex than a face tracker, but it does not learn and improve, it merely incorporates more data into a geometric model. Players can choose where to take photos based on what is convenient or what they are interested in, but they don't have much room for creativity beyond capturing the scene as it exists.

An example of high human agency and creativity is Zitnick and Parikh's database of abstract scenes [226]. Users construct a clipart scene of a sentence like "Jenny threw the beach ball angrily at Mike while the dog watches them both" and the result is many different interpretations of the same scenario. Although there is no machine involvement during the

scene creation process, the data is later analyzed to study high-level semantic information and which types of attributes are important for scene understanding.

Spiro's *Motion Chain* [182] is a webcam game for collecting example gestures for gesture recognition, and also allows for human creativity and agency, when players imitate the gestures of other players. The goal of collecting real-world data through a crowdsourcing game is similar to goal of *The Meme Quiz*, but in *Motion Chain*, the mechanics are not based on any gesture recognition system consuming the data, but on players observing and imitating each other.

In the opposite quadrant but also with a very similar goal, the AM-FED [133] facial expression crowdsourcing system has high machine involvement with low human agency. Human agency is low because the participants are asked to watch an advertisement and react the way they normally would if no camera were watching their expression. Machine involvement is fairly high, though, because after the advertisement is finished, the video of the user is quickly processed and analyzed with a face detector and smile tracker, and a chart showing when they smiled is shown to the user, along with a timeline of the video and an average smile graph across many users.

Deng's *Bubbles Game* [45] for bird classification is an interesting example of a game that technically doesn't have a machine or vision system running while the user plays the game, but where the player's actions directly influence how a computer vision system later makes its decisions. In *The Bubbles Game*, a user is guessing which species of bird is shown in an image, but the image starts out blurry and in greyscale. The user can place "bubbles" on the image that make that patch of the image in color and in focus, and the goal is to identify the bird with as few bubbles as possible. From this, the system learns what parts of the image or bird are most discriminative and useful for making its own classifications.

Although it is not a game, identity recognition and tag suggestion systems, like Facebook, have high machine involvement. They can make intelligent decisions based on context (who is likely to be in a photo based on friend relationships and who is often photographed together) and improve their capabilities over time as more faces are tagged. We consider these systems

to have low human agency, where there is an objective right answer of who is who in a photo, and the only user choice is whether or not to post or tag a photo in the first place.

6.3.4 Advantages of High Machine Involvement

We observed that the design space quadrant of high human agency and high machine involvement has certain advantages. Player agency is an important aspect of game design; according to Sid Meier, "a [good] game is a series of interesting choices." [157] Machine involvement, especially AI/ML/computer vision systems integrated into games in new ways, can create new types of experiences for players. Additionally, these systems can give designers a way to monitor the health of the data as its collected, instead of waiting until the game is over to process and evaluate the data. Conveniently, we found we could adapt the data collection task on the fly by adding more meme expressions without breaking the flow of the game.

6.4 The Meme Quiz

The goal of the game is to have the system correctly guess which facial expression the player is making. The player sees three meme expressions selected at random and chooses one to imitate. The player then takes a photo of herself imitating that expression. The system analyzes the photo and makes a guess, allowing the player to tell it the correct answer. Figure 6.4(a) shows the acting phase, and Figure 6.4(b) shows the game making its guess and asking for confirmation from the player. Behind the scenes, the system does not compare the new face directly to the meme, which could be a non-human animal or drawing, but to the faces of everyone who has acted out each of those three expressions before. Figure 6.4(b) shows the game displaying the top five most similar faces of other players, providing insight into the system's decision and why it might have gotten confused.

When the game was launched, the system had no training examples to learn from, so it guessed randomly. As play continued, there were more faces to compare with and it became easier to find a correctly-labeled similar-looking face. There were some specific ways the system could guess incorrectly, though. For example, the system occasionally thought that

a photo of a particular player looked most like that same player making a different expression. Or that the lighting was too different for the faces of otherwise close appearance and expression to seem similar. These are common mistakes for all facial expression recognizers to make, and the best solution is to collect more data to fully understand these distinctions.

6.4.1 Face Processing

Behind the scenes, the game runs through a standard facial expression recognition pipeline shown in Figure 6.5 and described below. First, we run the Face++ [83] face detector on the image to find the location of the face within the image and to find the location of landmarks such as eyes, nose, and mouth. Knowing the position of the landmarks, we align and crop the face to a common frame of reference where eyes, nose, and mouth are always in the same place, allowing us to easily compare different faces. Once we have the aligned and cropped face image, we turn that image into a lower-dimensional feature vector by computing Histogram of Gradients (HOG) features [39] in a grid over the face. This feature vector converts the pixel representation of the face to a representation of the shape and intensity of edges and lines in different regions of the face, which can capture changes in skin wrinkles and in mouth/eye/eyebrow shape while being impartial to skin tone and image color variations. This process of detecting, aligning, and computing low-level features to feed into comparison and classification tasks is the same used by Kumar et. al. in their work on searching within collections of faces [110] and on face verification [112].

After the face image has been transformed into a feature vector, we can compare it to other faces that have been processed in the same way. For a person imitating one of three memes, we look at all the faces also acting out those memes, find the face that is most similar to the new face, and set the system's guess to be the label for the closest face. Essentially, each face represents a point in N-dimensional space, and we are computing the new face's k-nearest neighbors (k-*NN* [36]) and checking their expression labels. The game makes its guess based on the single closest face.

As the game collects more data, there are more face points to compare to, filling out an

N-dimensional face space. During the game, we use feature vectors and a nearest neighbor classification algorithm. In our evaluation, we use these same feature vectors to train linear SVM classifiers. We did not use SVMs during the live game because $k-$NN was much simpler to implement; unlike SVMs, which would have required retraining after each new face or use of an online, incremental SVM.

6.5 Game Deployment

We built a website and an iOS application for *The Meme Quiz* and seeded the set of expressions with about twenty popular Internet memes. Over time, we expanded the set to 298 expressions, including memes suggested by users, expressive faces from movies, celebrities making extreme expressions, and the six basic expressions. Figure 6.7 shows the number of memes active in the game over time, and how adding new faces impacted performance, which we will discuss more in the next section.

198 players played 2860 rounds (one photo per round), averaging 14.4 rounds/photos per player. Two-thirds of the photos (1893 out of 2860) were captured through the iOS application. There were 110 women, 61 men, and 27 people who did not indicate a gender. Players spanned a wide age range: 13-15 years: 17 players, 16-19 years: 24 players, 20-29 years: 55 players, 30-39 years: 26 players, 40+ years: 5 players, and 7 players of unknown age.

We asked users for the country in which they were born, as well as the country in which they currently lived. There is a spread, and the most common responses for current country was: United States: 98 players, unknown/decline to state: 24 players, Mexico: 8 players, Brazil: 6 players, France: 5 players, Columbia: 4 players, UK: 4 players, Russia: 3 players, Turkey: 3 players, Iran: 3 players, and so on. For birth country, the top responses were: Unknown/decline to state: 133, United States: 44, Ukraine: 3, China: 2, Korea: 2, India: 2, and so on.

Ideally, we would like to have a broad and dense sampling of expressions across all gender, age, and ethnicity permutations.

Our consent form, which every player was required to consent to before playing, allowed players ages 13 and older to participate. Ethically, collecting faces for a public research database, especially one known to contain minors, is a touchy subject. In our defense, our subjects are explicitly told that they are contributing their face to an anonymized public research database, and can opt not to participate. One alternate approach is to perform research on privately-owned user data collected by Facebook, such as the 4,000-person, 4.4-million photo dataset used by Taigman et. al. [188]. A second alternate approach is to scrape the Internet for images of emoting faces, such as the dataset collected for the 2013 Kaggle Facial Expression Recognition challenge [70], which contains over 30,000 faces. In our analysis of this dataset, we discovered that around 20% of these images were duplicates, many were mislabeled, and many were stylized stock photographs of models overacting each expression. In both of these cases, there are thousands of people who have no idea their faces are in these datasets. We value transparency and would like to see it become more common to intentionally build and contribute to computer vision research datasets.

6.6 Evaluation

Our evaluation focuses on studying the system's learning process. Because of the large number of expressions included in the game, not all have enough examples to be usefully analyzed, so the evaluation in the following section will focus on the 26 expressions that have 30 or more examples. Figure 6.6 shows eight of these expressions and the average blend of imitations. We generated these average images by computing the average RGB pixel value for each pixel in a stack of faces. Conveniently, the face images are pre-aligned because of the preprocessing step to detect, align, and compute features, so eyes, mouth, and other landmarks line up. It is possible to compute an even sharper average blend using a technique called Collection Flow [93].

The three key questions we investigate in our evaluation are:

1. How well does the system learn over time?

2. Which expressions are learned best? Which expressions are confused?

3. How does our dataset compare to existing datasets and are we collecting useful data?

6.6.1 Learning Over Time

We made a game around teaching the system to recognize facial expressions, but with the challenges of so many expressions to master and so many possible facial appearances from "in the wild" players, did the system successfully learn? Instead of six basic expressions to master, the system was exposed to hundreds of new expressions (although only three at a time). It saw hundreds of different human players, each with different face shapes, hairstyles, accessories, and glasses. It was also exposed to many different lighting environments and shadows cast on faces in different directions. Would the game be able to make sense of the underlying expressions in spite of all these variations?

As the game deployment progressed, we tracked how often the system guessed correctly. Figure 6.7 shows a plot of accuracy over time. If the game were guessing randomly, it would have an accuracy of 33%, but the accuracy was significantly higher than that and increasing over time; over the course of the 2,860 rounds played, accuracy has progressively increased to 60.69%. The best results of a recent Kaggle competition on facial expression recognition [70] come in at 71.2% accuracy on six expressions.

At five points during deployment, we expanded the set of memes (these points are represented by the vertical bars in Figure 6.7). As can be seen, immediately following each expansion, accuracy takes a temporary dip. For example, when we added 185 new expressions, mostly celebrities making goofy faces and extreme expressions, accuracy dropped from 52% to 48% as the system had no information about these new faces. But over time, players imitated these expressions and the system was able to correctly guess them.

Being able to monitor the health of the system during the deployment and to verify that the game was actually improving at its expression recognition task was both essential and encouraging. Furthermore, our learning system was flexible enough to allow us to change

what it was learning on the fly, and to rebuild its own knowledge base when we gave it completely new challenges.

6.6.2 Which Expressions Learned Best/Worst?

With so many memes to recognize (298 by the end of the deployment), and certain memes that either showed the same expression or looked very similar, we wanted to know which expressions were being learned most reliably. Some expressions might do worse than others if they either had very little data, or if they looked too similar to other expressions.

Because the game only tests three expressions at a time, we were interested in evaluating the full knowledge of the system at the end of deployment. We focus on the top 26 expressions with 30 or more examples, ending up with 930 samples. Instead of the nearest-neighbor classification used online in the game, we fed these 930 faces into a 26-class SVM, which we then used to predict the expression label of a new face. We randomly split our dataset into training and testing sets (66% of the data in the training set, 33% in the remaining testing set) and repeated this training/testing ten times, then averaged the results.

Overall, expressions were correctly predicted 38.89% of the time. Note that this prediction is choosing from twenty-six labels (thus a baseline accuracy of 3.8%), not just three labels as in the game itself.

The confusion matrix shown in Figure 6.8 illustrates which expressions were correctly predicted and which were classified as different expressions. The dark diagonal shows where expressions were successfully classified. Dark squares off axis indicate expressions that were confused. The most frequently confused expressions are listed in Table 6.2. Many of these confusions make sense; Grumpy Cat and Not Bad Obama expressions both have a similar appearance of the lines around the mouth coming from Grumpy Cat's frown and Obama's mouth shrug. Borat and a picture of Ellen Page making "bunny teeth" also look similar and are understandably confused.

In the future, we imagine our data being used to distinguish between similar-looking expressions, e.g., learning to look at the eyebrows to differentiate "grumpy" and "not bad".

Meme A	Meme B	Percent
Borat	Bunny Teeth	21.5%
Bunny Teeth	Borat	21.2%
Borat	Overly Attached Girlfriend	19.8%
About to Interrupt	Bro Paul Ryan	19.2%
Crazy Eyes Nick Cage	Happy Kenneth	17.4%
Liz Lemon Eye Roll	Grumpy Cat	17.1%
No.	Not Bad Obama	16.8%
Grumpy Cat	Not Bad Obama	16.6%
Mr. Bean	Zooey Kiss	16.6%
OMG Ribbons cat	Einstein Tongue	16.2%
No.	Grumpy Cat	15.9%
Crazy Eyes Nick Cage	Overly Attached Girlfriend	14.6%

Table 6.2: Expressions commonly confused with other expressions over 14% of the time.

We intend to perform a hierarchical classification of expressions, and compare it to the six basic expressions and the FACS basis space of expressions based on face muscles.

6.6.3 Comparison with CK+

The facial expression dataset from Cohn-Kanade (CK+) [124] is a standard in facial expression recognition research, capturing 127 subjects making the six basic expressions of joy, sadness, anger, fear, surprise, and disgust. We included photos of people making these expressions in our prompt set to directly compare with CK+.

For the comparison we used a subset of our data – the 317 photos with the six expressions in CK+ – and compared it to the 327 photos of these expressions in CK+, not including neutral expressions. For each dataset, we trained a classifier for each expression, and then classified faces from the opposite dataset. We found that when training classifiers using CK+

data, we were only able to recognize 48.58% of *Meme Quiz* faces. In contrast, when we use *Meme Quiz* data as the training set, we are able to recognize 75.84% of CK+ faces. This means that CK+ data is not diverse enough (many of the faces have the same lighting) to serve as training data for real world scenarios such as ours. In contrast, this small subset of our data, with its real-world faces captured in diverse lighting environments, still works to predict standard examples of these expressions fairly reliably.

6.7 Limitations and Future Work

6.7.1 Verification

The goodness of our system relies on (most) players making their best effort to imitate each expression and providing the correct answer at the end of each quiz round. The goal of the game, whether or not players choose to play this way, is to teach the system enough about each expression to have it correctly guess as often as possible. However, players can lie, or they can game the system and attempt to teach the system incorrect concepts. We have no explicit verification method for dealing with scenarios like this, and can only hope there is enough good data to treat misinformation as outliers.

We could use humans to verify each others' faces by having them act as the 'computer' and guess which expressions other players are making. This could give us useful information about which expressions are incorrectly labeled, which players are not good actors, or which players are not good at reading expressions.

An alternate, partially automated solution would be to use the trained expression classifiers to look for outliers and present those to a human for verification (either inside or outside of the game context). Essentially, the game's technology has a built-in tool for identifying the most confusing or questionable faces, which would simplify the verification task for the human.

6.7.2 Engagement

We struggled with questions of how to engage users. What would make people want to take photos of themselves? What would make them keep playing? Could we generate a compelling artifact, e.g., a picture of their face swapped with a meme, that they could share with their friends to draw new users to the site? Would the idea of "teaching the computer" be compelling enough, or would it be too nerdy?

We tried many different approaches, including generating swaps and sharable artifacts (which did draw new people to the site) but we were not rigorous enough in our studies to draw any conclusions. However, despite the numerous surface changes we made, the core mechanic of *The Meme Quiz* remained constant, and the system kept on learning.

6.8 CDCV Properties and Outcomes in The Meme Quiz

The Meme Quiz represents a more generalized application of the CDCV framework than PhotoCity or PointCraft. In PhotoCity, every design decision is tied to the specific 3D reconstruction pipeline. The Meme Quiz, on the other hand, represents how a CDCV system can be built around a computer vision system that makes predictions, e.g., the inference stage of a machine learning algorithm. This likely makes it the project that is most transferrable to a new problem domain.

The two CDCV properties that this project explores most deeply pertain to how the core computer vision technology is used and the interactivity afforded by the game. The computer vision system of detecting faces and predicting expressions forms the core mechanic of the game and incorporates the uncertainty of the system into how the user interacts with the system. The interactivity also provides a low barrier to entry, running on any device with a camera and web access, and making it easy to contribute and receive immediate feedback.

The two properties that are explored more shallowly are those of the mappable data space and the feedback explaining contributions in terms of that map. While capturing diverse expressions under real world conditions provides overall motivation for the project, progress

towards this mapping goal is not visualized to the user, partly due to privacy concerns. The feedback provided by the game somewhat explains the computer's guess by showing the five nearest faces, but not how the user's new data impacts the map overall. Ideally, there would be feedback that shows how the user's new photo fits into the existing mappable space and what effect it has on updating the map.

In terms of CDCV outcomes, The Meme Quiz let users contribute data for the memes that interested them and provided a channel for submitting new memes. The dataset collected by the Meme Quiz is the only one of its kind; there is no other source of data of people directly mimicking diverse expressions, although there are datasets of diverse expressions that are annotated with which expressions appear similar [204]. The inherent diversity of users' facial appearances contributed to the diversity of the dataset. Participants found creative ways to better match their faces to the memes by changing their poses, hair, and even makeup. The Meme Quiz provided an approachable and engaging platform for collecting this data.

Outside of the defined CDCV system properties, an additional aspect of the Meme Quiz that encouraged participation was that of a shareable artifact; for every face a user imitated, the system generated an image with the two faces swapped, which the user could share online. These shareable artifacts provided a way for users to demonstrate their contributions to their friends, and by doing so, attract new users to the system.

6.9 Conclusion

We have presented *The Meme Quiz*, a game for teaching computers to recognize facial expressions by having players imitate Internet memes and quiz the computer. We have successfully evaluated the learning process of our game, confirming that, despite the wide variety of expressions and faces it was seeing, it did improve over time, even when we adapt the game to include more memes. Furthermore, the expressions it confused were often similar-looking and point to strategies one could employ in the future to help distinguish similar expressions. Compared to existing datasets of facial expressions, our data are more diverse and potentially provide a better training basis than the existing datasets themselves.

We have situated our game in a design space of data-gathering games, which we have split along two axes of *human agency* and *machine involvement*. Our system has high human agency with players acting out expressions of their choice, as well as high system involvement, with a face recognition system evaluating each face and learning as it goes. Observed benefits of being in the high-human-agency, high-machine-involvement quadrant include being able to monitor the health of our data over time, as well as make adaptations to the game without impeding the system's learning process.

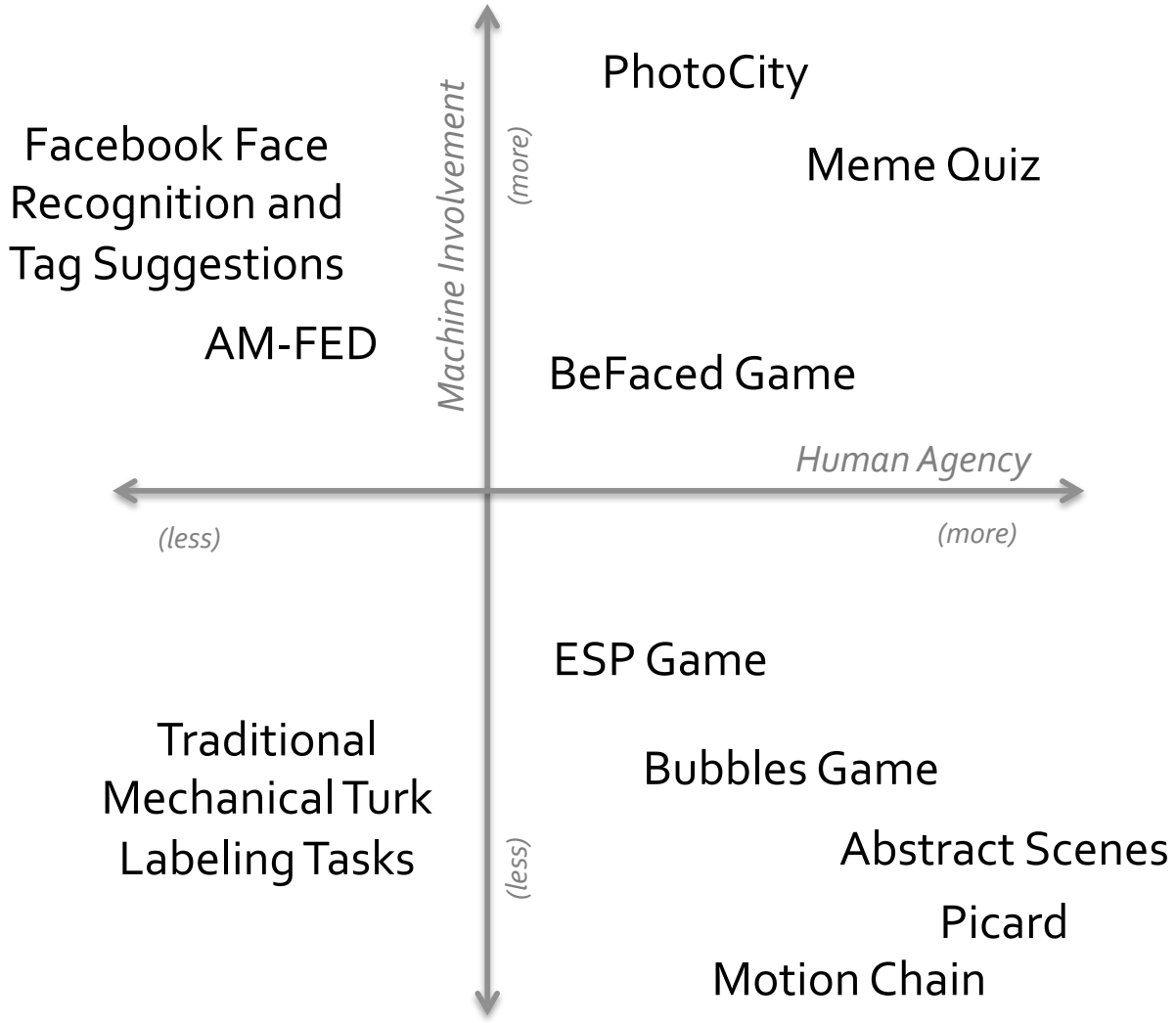
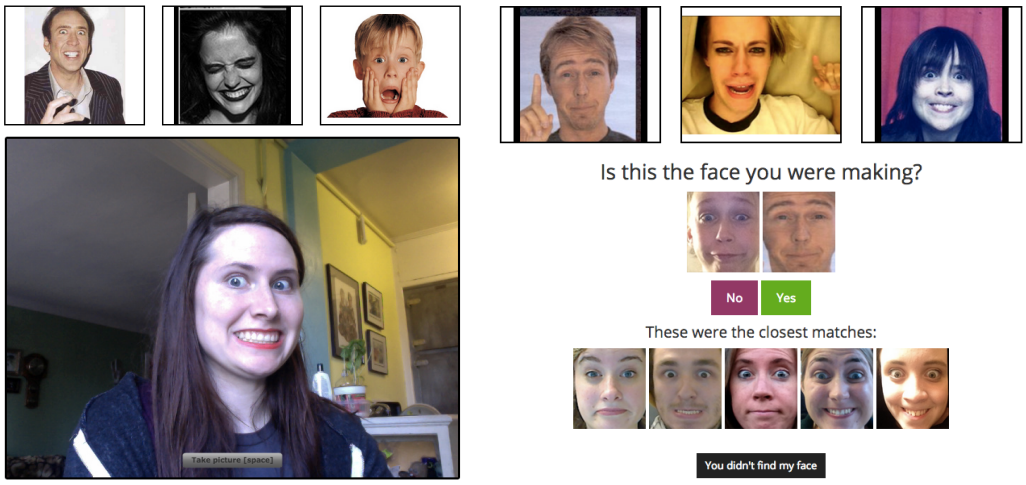


Figure 6.3: Design space of data-gathering and data-generation games (and other systems). Does the human get to make choices and express creativity (high agency), or is there ultimately one objective right answer the human is expected to provide (low agency)? Is there an automated system that processes user data and is in the domain in which the data is expected to be used (high machine involvement)? Does new data change the way the system operates, and does the system learn or improve over time?



(a) Imitating an Expression (b) Verifying the Game's Guess

Figure 6.4: The quiz interface: (Left) At the top, there are three possible expressions to imitate. Below, the user sees a live preview of her own face and can press a button to take the photo. (Right) After the player takes a photo, the game system makes its guess and shows its answer to the player, who then provides the correct answer. In this example, the game (correctly) guessed the left-most expression, but the top five closest faces include three from the (incorrect) right-most expression, suggesting that this was a difficult example.

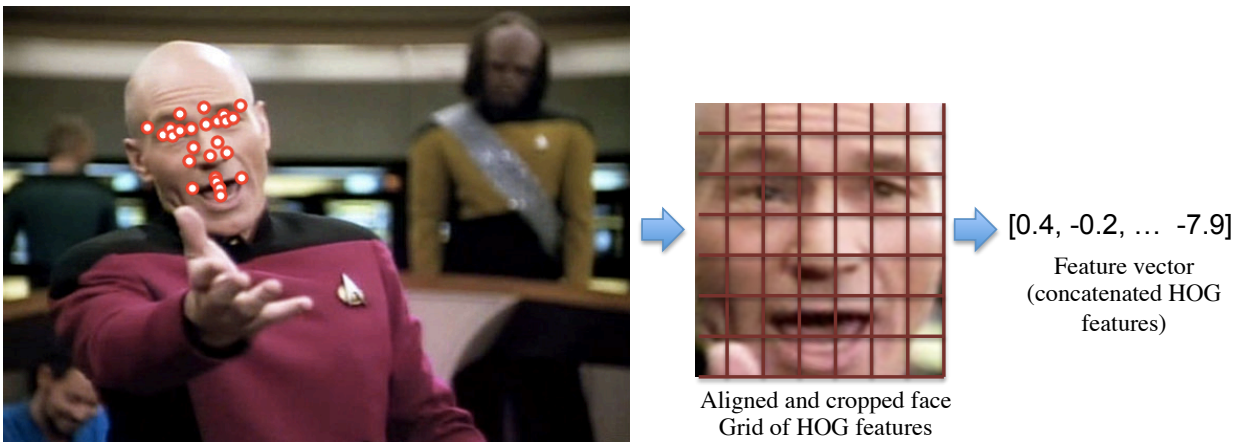


Figure 6.5: Every face that is uploaded to the game goes through this pipeline. First, a face detector finds locations of landmarks on the face, such as eyes, eyebrows, nose, and mouth. Using those landmark locations, we align the face to a common reference frame and crop the image to just the face region. Then we compute a grid of Histogram of Gradient (HOG) features and concatenate them together to get our final feature vector that we use for classification.

(a) Not-Impressed-McKayla (57 photos)

(b) Overly Attached Girl (43 photos)

(c) Grumpy Cat (43 photos)

(d) OMG Ribbons Cat (42 photos)

(e) Not Bad Obama (42 photos)

(f) Angelina Fish Lips (41 photos)

(g) 'No.' Rage Face (40 photos)

(h) Zooey Kiss (39 photos)

Figure 6.6: Example expressions and their average blends.

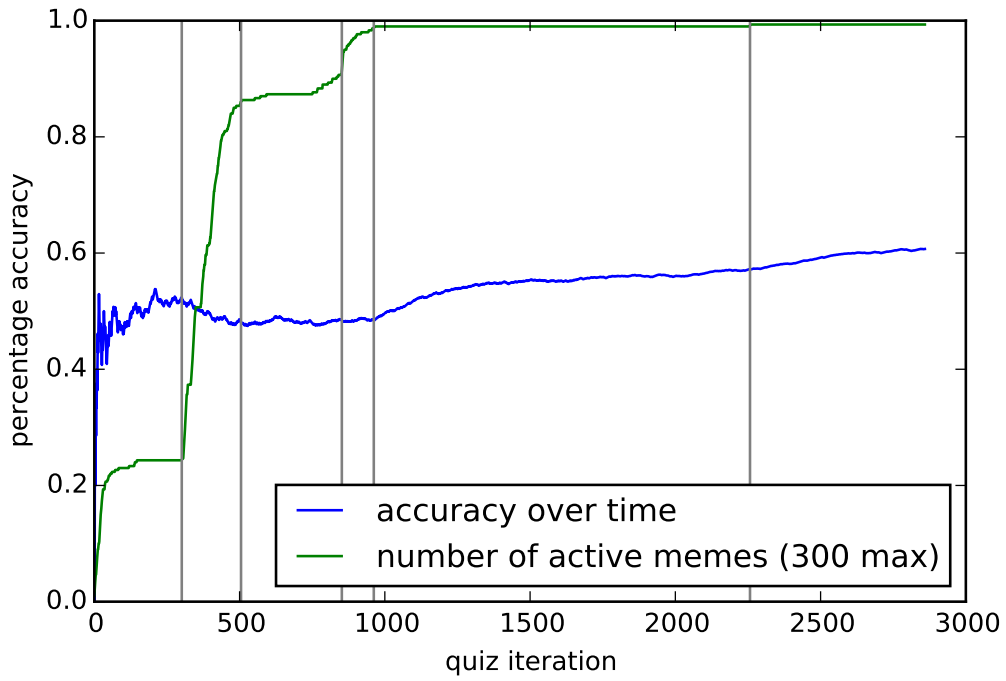


Figure 6.7: Over almost 3,000 quiz iterations, the game's accuracy (shown in blue) in choosing the correct expression out of three climbs from 33% (random) to above 60%. With more quiz rounds, this number would likely continue to increase. The green line shows how many meme expressions are active in the game, with the vertical bars indicating when more memes were added. Adding new memes, which the system knows nothing about, temporarily hurts performance until the game collects enough examples to learn those memes.

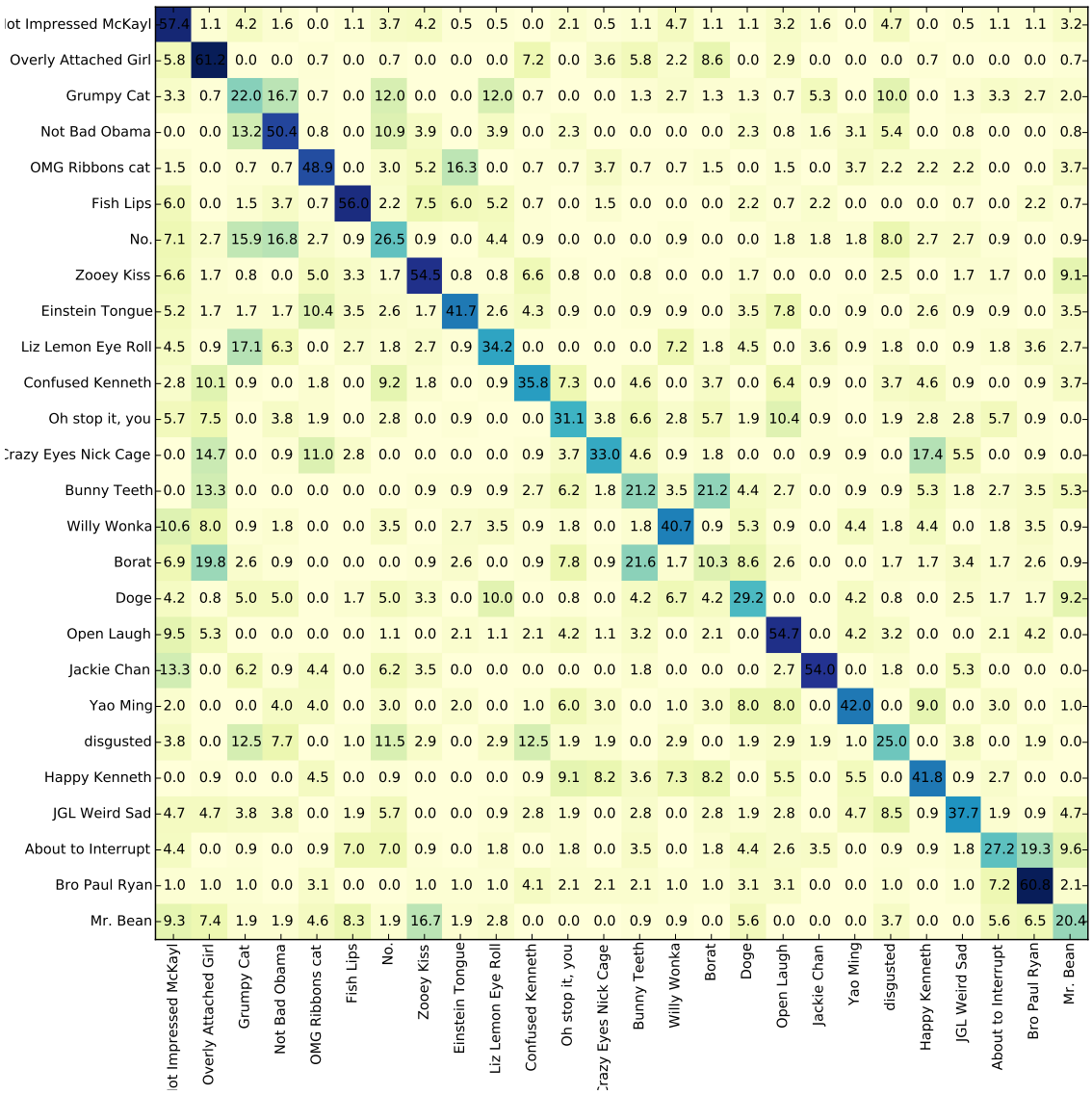


Figure 6.8: We trained and tested (using cross-validation) a multi-class SVM on the 26 expressions. (Each expression has between 30 and 57 training examples.) This confusion matrix shows which expressions were commonly categorized as other expressions, such as Grumpy Cat with Not Bad Obama. The confusion matrix is not exactly symmetric.

Chapter 7

CONCLUSION

7.1 Contributions

My work brings computer vision and crowdsourcing together in a way that puts the humans involved in crowdsourced work in the driver's seat in order to steer their efforts towards solving problems they care about. When computer vision research gets applied in the real world, there are issues of limited training data, batch algorithms that are slow to run and reprocess, and limited ways of interacting with the algorithms and the data. My work aims to hasten the successful application of computer vision technology in a wide variety of applications by calling for interactively designed systems that enable rapid collection and testing of new data by distributed crowds of people, in ways that the crowds have collective agency to collaborate and innovate. The key is introducing a new structure of computer vision systems that makes them platforms of distributed innovation.

There are three fields of study that this work spans: computer vision, human-computer interaction (HCI), and game design. This work requires understanding how specific computer vision systems work, how the data for these systems is sourced including how it is annotated at scale through crowdsourcing, and the role crowdsourcing has played in rapidly advancing the field. This work also requires understanding from an HCI perspective how crowdsourcing itself has been a subject of research, what scalable crowdsourcing looks like, what is possible with crowd pipelines based on microtasks, and what are alternative crowdsourcing systems for organizing and motivating people that allow for new possibilities, especially in the realm of groundbreaking collective intelligence and creativity. The final piece of the puzzle is understanding game design, both how it has been used for crowdsourcing purposes through games with a purpose (GWAPs) but also its decades of depth and insight for designing

interactive systems that individuals or groups of people use, and how those systems drive motivation, engagement, learning, creativity, and complex system understanding.

The primary contribution of this work is the development of the crowd-driven computer vision framework, which combines ideas from these three fields into design recommendations for building new interactive crowdsouricng systems that advance the capabilities of computer vision. The CDCV framework defines four properties of CDCV systems including the fact that there is a central computer vision technology, there is a mappable space of data pertaining to that technology, the CDCV system makes that space interactively mappable by a crowd, and that users receive meaningful feedback from interacting with and contributing to the CDCV system. I claim that the properties of this framework, when applied to the design of a crowdsourcing system for computer vision, support certain outcomes including (1) accumulating data in areas of user interest, (2) the possibility of collecting new types of datasets, (3) engaging the creativity, skill, and diversity of the crowd, and (4) motivating and focusing crowd effort towards a shared purpose. These outcomes can, in turn, contribute to the development of computer vision systems that are more useful, robust, effective, and fair. I demonstrate this framework through three experimental CDCV systems that elicit these outcomes. The three systems described in this dissertation are PhotoCity, PointCraft, and the Meme Quiz.

In PhotoCity, I built an game around a structure from motion 3D reconstruction pipeline that enabled distributed computing and distributed involvement from a crowd of people. In particular, the game enabled a large (100K) dataset of images with dense coverage of an area to be collected by a relatively small (<40) group of people in a relatively short (6 weeks) amount of time. The coverage of the photos represents a density and connectivity not represented in online photo collections like those found on Flickr. This was enabled through an interactive and engaging game that provided points and feedback for each individual contributed image (not just a quantitative number of images) such that players were able to learn from this feedback and develop strategies to perform better in the game, and to understand a bit about the underlying computer vision without having been explicitly taught

about it. This project represents using localized knowledge and access of local players on the ground to take photos from vantage points they can get to but that also haven't been captured before. It represents the players building up the computational representation of their real world environment (akin to teaching the system) through repeated interactions and contributions. It also represents the players learning about the underlying system and how to be more efficient in it through being able to interact quickly and experiment and get rapid, fine grained feedback. These are the makings of the crowd-driven computer vision framework, which further projects explore specific areas more in depth.

My next project, PointCraft, is a deep dive into utilizing human skills of perception, manipulation, and construction to address a problem that computers are not currently able to solve satisfactorily. The photos from PhotoCity are used to generate 3D models in the form of dense point clouds. But those point clouds are unable to be turned into high quality, clean, polygonal geometry of the kind an artist would manually create for a videogame or AR/VR application. The point clouds are noisy (the points don't always lie on flat surfaces with each other), they have gaps and holes due to photos not existing at certain angles, or objects occluding other objects (like foliage in front of buildings), and they have reconstruction errors where geometry is doubled or misaligned due to how the structure from motion algorithm works. To a human, these point clouds contain more than enough information about the buildings they represent, especially given our human experiences with seeing and understanding buildings throughout our lives, and especially for the humans that have seen the physical versions in real life of these digital recreations. With PointCraft, I sought to build a 3D reconstruction tool, inspired and borrowing interaction techniques from a well known 3D construction game, that would let human players interpret and transform these rough, incomplete structures into clean, complete, usable geometry. The result is a tool that enables the creation of 3D geometry in a situation that is very challenging for computes, but not very challenging for humans, given the right tools.

The final project is the Meme Quiz, a project in the realm of facial expression recognition that aims to address issues of representation in facial recognition software, specifically

in emotions themselves which are often reduced to just six emotions. This project creates a flexible framework for collecting images of a wide variety of people making many different communicative or emotive facial expressions in a variety of real world environments. Furthermore, this project explores the idea of letting a crowd of people teach a computational system in a way that improves the system's abilities for everyone.

For each of the three fields from which crowd-driven computer vision was influenced, there is a message or contribution back to that field. In computer vision, the idea is that new types of datasets are needed to drive new applications, and we need new ways of collecting that data. The way of collecting new types of data that I propose is by building interactive and crowd-driven systems than many people can collaborate on together and involve the end users of the system in the design and development of the system itself. In HCI, my work is an extension of work on crowd pipelines that go beyond microtasks and use deep human expertise and computer supported cooperation as well as training experts as they go along. In game design, my work is an extension of games with a purpose, to, similar to crowdsourcing, involve human expertise and skill in a deep way. Additionally, my work (PhotoCity and the Meme Quiz in particular) is the first to use computer vision actively within a game with a purpose as a core mechanic of the game.

7.2 Future Work

I propose three areas of future work in crowd-driven computer: depth, breadth, and support tools to aid the adoption of CDCV ideas by new people.

My first proposal is to explore the depth of CDCV by applying ideas from the framework to many more phases of the structure-from-motion pipeline. In PhotoCity, the only part of this pipeline we opened up to crowd interaction was the incremental expansion of 3D models. As we exposed more information about the underlying system to users, e.g., through the visualizations afforded by PointCraft, users were more able to understand and identify the mistakes made by the technical system. These mistakes included the incorrect registration of a single photo that then caused subsequent photos to be positioned incorrectly and errors

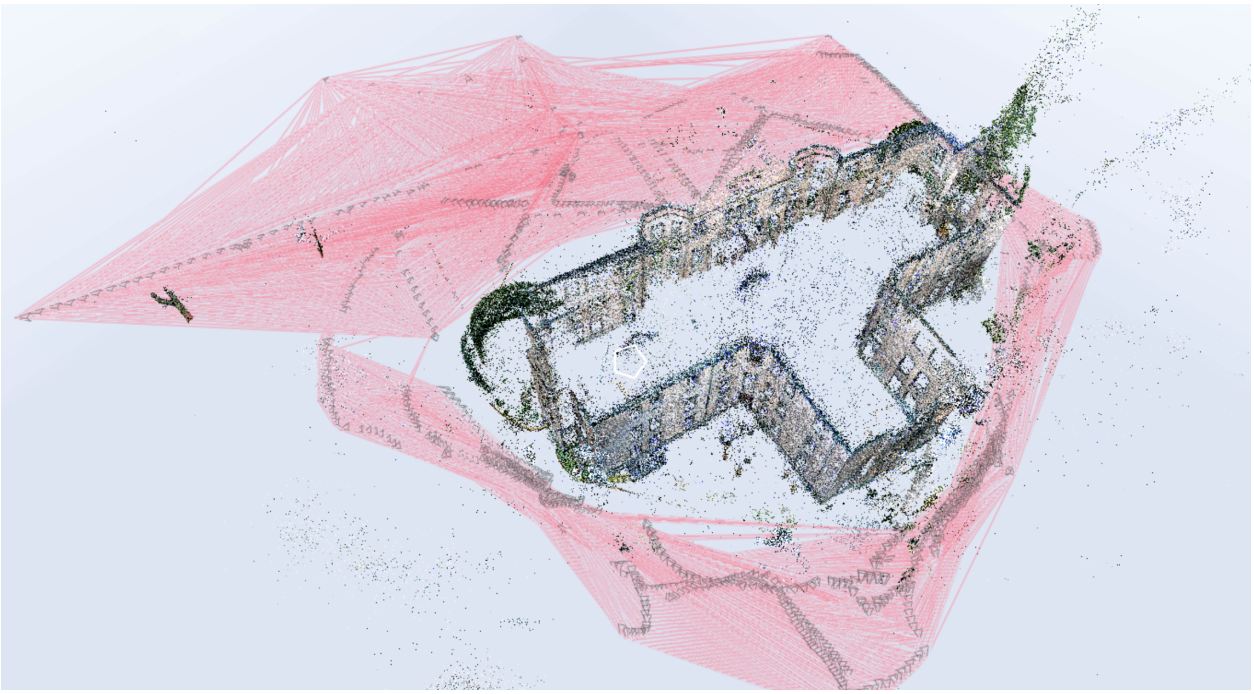


Figure 7.1: This image shows an overhead view of the Lewis Hall point cloud from PhotoCity, rendered with PointCraft, which was modified to show the positions of the cameras and the visual connectivity of these cameras. The pink lines correspond to photographs have visual correspondences that are used to generate the 3D geometry. On the left of the image, there are many photographs that view the same information and could be matched together to construct more consistent geometry.

to accumulate, or the fact that when a building was constructed the full way around, there would be loop-closure issues due to which images were understood to be connected to which other images as shown by the pink lines in Figure 7.1, and the geometry would not match up. The human users could easily identify these errors, but the tools did not yet exist to allow them to fix these errors and drive these systems out of trouble. There were many places in the structure-from-motion pipeline specifically that relied entirely on automated assumptions (e.g., which photos were compared with one another, what specific pixel-level correspondences were found). While this automation is important for computing at scale, it is fallible and ought to have mechanisms for identifying and overcoming those failures. There

is an opportunity to elicit human feedback at this level in order to collect data about the kinds of mistakes being made by the automated techniques and how to correct them.

My second proposal is the broad application of the CDCV framework to other areas of machine perception outside of computer vision such as natural language processing, conversational agents, or human-robot interaction. Systems in these areas rely on training data, but this data is hard to find and hard to make because there are many nuances and edge cases to capture, especially around how humans interact with these systems in unexpected ways. By applying ideas from this dissertation, it will be possible to crowdsource the collection of new datasets by building crowd-driven interactive systems that let users experience and experiment with these applications, and get feedback about their contributions, and understand how they are collaboratively improving the technology they want to use.

My third proposal is to develop design resources and software tools to aid the adoption of CDCV ideas and to help others apply the CDCV framework to their own work. I imagine these design resources taking the form of design guidelines similar to Apple's Human Interface Guidelines for Machine Learning.[1] I imagine software toolkits or libraries that make it easy to build and deploy CDCV systems similar to how MediaWiki [9] embodies the interaction model of the Wiki concept [116] that can be repurposed in many new domains. Some of the particular challenges of designing CDCV systems that are shared across domains involve figuring out how to provide intuitive feedback to users, how to motivate users, how to store user-contributed data, how to protect against malicious users, and how to be responsible with that data. These resources and software tools would allow lessons learned in one CDCV application to benefit new CDCV applications. The learnings would encompass the four properties of crowd-driven computer vision: integrating with a wide variety of computer vision systems, conveying the mappable data space to users even when the data is non-spatial, designing interactions that let users efficiently and effectively communicate their intent, and designing intuitive feedback that clearly conveys to users the impact of their contributions.

---

[1] `https://developer.apple.com/design/human-interface-guidelines/machine-learning/overview/introduction/`

7.3 Epilogue

This work was initially borne out of mild disillusionment with research technology that worked beautifully in some cases but failed in others. My investigation into why often lead to an issue with the data—there wasn't enough data or the data represented a particularly challenging scenario for the technology. In parallel, I had been working on the crowdsourced protein folding game Foldit and on my own projects involving collaborative creativity and emergent behavior. In those, I saw the opposite of disillusionment: surprise, delight, and awe at the depth and creativity of the users of these systems, especially as they learned from each other and built on each other's work. It made sense to combine these ideas together, to apply the wisdom, creativity, and collaboration of the crowd to solve data challenges underlying so many computer vision and machine perception problems. Since starting down this path, I have transitioned from thinking not just about *how* to do this, but *why* it is important.

The data that goes into computer vision systems, especially the human labeling of this data, is often overlooked. In recent years, especially since 2017, there is more discussion about how datasets are sourced, how those sourcing methods impact bias, and what the implications of those biases are, some of which I mention in Chapter 2. There is also increasing discussion of the invisible labor that goes into computing, labor that has historically always been present and nearly always overlooked [77, 56]. This is true today with the proliferation of artificial intelligence and machine learning applications that use humans to produce, annotate, filter, and validate much of the input that feeds these data-hungry systems. My work acknowledges that these systems need human-curated data and human guidance, but that this should be a transparent process that the end-users of these systems can be involved with and can steer themselves to build applications that directly benefit the end-users.

I see my work as in alignment with Keyes' call for systems and spaces that support autonomy and dignity [97]. In the context of computer vision, I take autonomy to mean having control over one's own data and being able to train and use computer vision for one's own purposes without the only applications available being the ones deemed economically

viable by a large institution. As for dignity, to support dignity would be to hear the concerns of users about the undue risks and potential harm they face through technology, even if it is hard for the application developers to understand those risks applied to themselves, and to take steps to mitigate those risks.

This dissertation is part of a larger discussion about the role of computers to augment human abilities, steps taken throughout history to democratize access to computing, and how to identify the truly beneficial applications of technology to problems facing society.

BIBLIOGRAPHY

[1] Sameer Agarwal, Noah Snavely, Ian Simon, Steven M Seitz, and Richard Szeliski. Building rome in a day. In *2009 IEEE 12th international conference on computer vision*, pages 72–79. IEEE, 2009.

[2] Nina Amenta and Marshall Bern. Surface reconstruction by voronoi filtering. In *Proc. of the fourteenth annual symposium on Computational geometry*, SCG '98, pages 39–48, New York, NY, USA, 1998. ACM.

[3] Nina Amenta, Sunghee Choi, and Ravi Krishna Kolluri. The power crust. In *Proc. of the sixth ACM symposium on Solid modeling and applications*, SMA '01, pages 249–266, New York, NY, USA, 2001. ACM.

[4] Apple. Understanding world tracking in arkit. `https://developer.apple.com/documentation/arkit/understanding_world_tracking_in_arkit`. Accessed: 2019-04-12.

[5] Cecilia R Aragon, Sarah S Poon, Andrés Monroy-Hernández, and Diana Aragon. A tale of two online communities: Fostering collaboration and creativity in scientists and children. In *Proceedings of the seventh ACM conference on Creativity and cognition*, pages 9–18. ACM, 2009.

[6] Jostin Asuncion. Know your meme: Reaction images. `http://knowyourmeme.com/memes/reaction-images`. Accessed: 2015-04-24.

[7] Chris Baber, James Cross, Tariq Khaleel, and Russell Beale. Location-based photography as sense-making. In *BCS-HCI '08: Proceedings of the 22nd British HCI Group Annual Conference on People and Computers*, pages 133–140, Swinton, UK, UK, 2008. British Computer Society.

[8] Tadas Baltrušaitis, Peter Robinson, and Louis-Philippe Morency. 3d constrained local model for rigid and non-rigid facial tracking. In *2012 IEEE Conference on Computer Vision and Pattern Recognition*, pages 2610–2617. IEEE, 2012.

[9] Daniel J Barrett. *MediaWiki: Wikipedia and beyond.* " O'Reilly Media, Inc.", 2008.

[10] Marek Bell, Stuart Reeves, Barry Brown, Scott Sherwood, Donny MacMillan, John Ferguson, and Matthew Chalmers. Eyespy: supporting navigation through play. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, pages 123–132. ACM, 2009.

[11] Sean Bell and Kavita Bala. Learning visual similarity for product design with convolutional neural networks. *ACM Transactions on Graphics (TOG)*, 34(4):98, 2015.

[12] Sean Bell, Paul Upchurch, Noah Snavely, and Kavita Bala. Opensurfaces: A richly annotated catalog of surface appearance. *ACM Transactions on Graphics (TOG)*, 32(4):111, 2013.

[13] Fausto Bernardini, Joshua Mittleman, Holly Rushmeier, Claudio Silva, Gabriel Taubin, and Senior Member. The ball-pivoting algorithm for surface reconstruction. *IEEE Transactions on Visualization and Computer Graphics*, 5:349–359, 1999.

[14] Michael S. Bernstein, Greg Little, Robert C. Miller, Björn Hartmann, Mark S. Ackerman, David R. Karger, David Crowell, and Katrina Panovich. Soylent: A word processor with a crowd inside. In *UIST '10*, UIST '10, pages 313–322, New York, NY, USA, 2010. ACM.

[15] Pravin Bhat and Sebastian Burke. Photospace: a vision based approach for digitizing props. In *SIGGRAPH Talks*, page 1, 2011.

[16] Jeffrey P Bigham, Chandrika Jayant, Andrew Miller, Brandyn White, and Tom Yeh. Vizwiz:: Locateit-enabling blind people to locate objects in their environment. In *2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition-Workshops*, pages 65–72. IEEE, 2010.

[17] Steve Branson, Catherine Wah, Florian Schroff, Boris Babenko, Peter Welinder, Pietro Perona, and Serge Belongie. Visual recognition with humans in the loop. In *ECCV'10: Proceedings of the 11th European conference on Computer vision: Part IV*. Springer-Verlag, September 2010.

[18] Joy Buolamwini and Timnit Gebru. Gender shades: Intersectional accuracy disparities in commercial gender classification. In *Conference on Fairness, Accountability and Transparency*, pages 77–91, 2018.

[19] Adrienne Burke. Games that solve real problems: Crowdsourcing biochemistry. `http://www.forbes.com/sites/techonomy/2011/10/27/games-that-solve-real-problems-crowdsourcing-biochemistry/`, 2011.

[20] Andrew Borg Cardona, Aske Walter Hansen, Julian Togelius, and Marie Gustafsson Friberger. Open Trumps, a Data Game. In *FDG '14: Foundations of Digital Games*, April 2014.

[21] Shan Carter, Zan Armstrong, Ludwig Schubert, Ian Johnson, and Chris Olah. Activation atlas. *Distill*, 4(3):e15, 2019.

[22] Irene Celino. Location-based games for citizen computation. In Pietro Michelucci, editor, *Handbook of Human Computation*, pages 297–316. Springer New York, 2013.

[23] Siddhartha Chaudhuri, Evangelos Kalogerakis, Leonidas Guibas, and Vladlen Koltun. Probabilistic reasoning for assembly-based 3d modeling. *ACM Trans. Graph.*, 30:35:1–35:10, August 2011.

[24] Anne-Laure Chauve, Patrick Labatut, and Jean-Philippe Pons. Robust piecewise-planar 3d reconstruction and completion from large-scale unstructured point data. In *CVPR*, pages 1261–1268. IEEE, 2010.

[25] Jenova Chen. Flow in games (and everything else). *Communications of the ACM*, 50(4):31–34, 2007.

[26] Neva Cherniavsky, Ivan Laptev, Josef Sivic, and Andrew Zisserman. Semi-supervised learning of facial attributes in video. In *ECCV'10: Proceedings of the 11th European conference on Trends and Topics in Computer Vision*. Springer-Verlag, September 2010.

[27] Lydia B Chilton, Greg Little, Darren Edge, Daniel S Weld, and James A Landay. Cascade: crowdsourcing taxonomy creation. In *CHI '13: Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. ACM Request Permissions, April 2013.

[28] Aidan Chopra. *Google SketchUp for Dummies*. John Wiley & Sons, 2007.

[29] A. Colburn, A. Agarwala, A. Hertzmann, B. Curless, and M.F. Cohen. Image-based remodeling. *Visualization and Computer Graphics, IEEE Transactions on*, 19(1):56–66, 2013.

[30] Katharine Compton. *Casual Creators: Defining a Genre OF Autotelic Creativity Support Systems*. PhD thesis, University of California Santa Cruz, 2019.

[31] Seth Cooper. *A Framework for Scientific Discovery through Video Games*. PhD thesis, Citeseer, 2011.

[32] Seth Cooper, Firas Khatib, and David Baker. Increasing Public Involvement in Structural Biology. *Structure/Folding and Design*, 21(9):1482–1484, September 2013.

[33] Seth Cooper, Firas Khatib, Ilya Makedon, Hao Lu, Janos Barbero, David Baker, James Fogarty, and Zoran Popovic. Analysis of social gameplay macros in the foldit cookbook. In *FDG '11: Proceedings of the 5th International Conference on Foundations of Digital Games*, Bordeaux, France, 2011. ACM Press.

[34] Seth Cooper, Firas Khatib, Adrien Treuille, Janos Barbero, Jeehyung Lee, Michael Beenen, Andrew Leaver-Fay, David Baker, Zoran Popović, et al. Predicting protein structures with a multiplayer online game. *Nature*, 466(7307):756, 2010.

[35] Seth Cooper, Adrien Treuille, Janos Barbero, Andrew Leaver-Fay, Kathleen Tuite, Firas Khatib, Alex Cho Snyder, Michael Beenen, David Salesin, David Baker, and Zoran Popović. The challenge of designing scientific discovery games. In *Proceedings of the Fifth International Conference on the Foundations of Digital Games*, FDG '10, pages 40–47, New York, NY, USA, 2010. ACM.

[36] Thomas Cover and Peter Hart. Nearest neighbor pattern classification. *Information Theory, IEEE Transactions on*, 13(1):21–27, 1967.

[37] David Crandall, Lars Backstrom, Daniel Huttenlocher, and Jon Kleinberg. Mapping the world's photos. In *Proc. Int. World Wide Web Conf.*, 2009.

[38] Chris Crawford. *The Art of Interactive Design.* A Euphonious and Illuminating Guide to Building Successful Software. No Starch Press, 2002.

[39] Navneet Dalal and Bill Triggs. Histograms of oriented gradients for human detection. In *Computer Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Computer Society Conference on*, volume 1, pages 886–893. IEEE, 2005.

[40] Matthias Dantone, Juergen Gall, Gabriele Fanelli, and Luc Van Gool. Real-time facial feature detection using conditional regression forests. In *2012 IEEE Conference on Computer Vision and Pattern Recognition*, pages 2578–2585. IEEE, 2012.

[41] James Davis, Joan Arderiu, Henry Lin, Zeb Nevins, Sebastian Schuon, Orazio Gallo, and Ming-Hsuan Yang. The hpu. In *Computer Vision and Pattern Recognition Workshops (CVPRW), 2010 IEEE Computer Society Conference on*, pages 9–16. IEEE, 2010.

[42] Paul E Debevec. Facade: modeling and rendering architecture from photographs and the campanile model. In *ACM SIGGRAPH 97 Visual Proceedings: The art and interdisciplinary programs of SIGGRAPH'97*, page 254. ACM, 1997.

[43] Paul E. Debevec, Camillo J. Taylor, and Jitendra Malik. Modeling and rendering architecture from photographs: a hybrid geometry- and image-based approach. In *Proc. of the 23rd annual conf. on Computer graphics and interactive techniques*, SIGGRAPH '96, pages 11–20, New York, NY, USA, 1996. ACM.

[44] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pages 248–255. Ieee, 2009.

[45] Jia Deng, J Krause, and Li Fei-Fei. Fine-Grained Crowdsourcing for Fine-Grained Recognition. In *Computer Vision and Pattern Recognition (CVPR), 2013 IEEE Conference on*, pages 580–587. IEEE Computer Society, 2013.

[46] Tamara Denning, Adam Lerner, Adam Shostack, and Tadayoshi Kohno. Control-Alt-Hack: the design and evaluation of a card game for computer security awareness and education. In *CCS '13: Proceedings of the 2013 ACM SIGSAC conference on Computer & communications security*. ACM Request Permissions, November 2013.

[47] Sebastian Deterding, Miguel Sicart, Lennart Nacke, Kenton O'Hara, and Dan Dixon. Gamification. using game-design elements in non-gaming contexts. In *Proceedings of the 2011 annual conference extended abstracts on Human factors in computing systems*, pages 2425–2428. ACM, 2011.

[48] Werner Dietl, Stephanie Dietzel, Michael D Ernst, Nathaniel Mote, Brian Walker, Seth Cooper, Timothy Pavlik, and Zoran Popović. Verification games: making verification fun. In *FTfJP '12: Proceedings of the 14th Workshop on Formal Techniques for Java-like Programs*. ACM Request Permissions, June 2012.

[49] Mira Dontcheva, Elizabeth Gerber, and Sheena Lewis. Crowdsourcing and creativity. In *CHI 2011: Crowdsourcing Workshop*, volume 10, page 4, 2011.

[50] Hao Du, Peter Henry, Xiaofeng Ren, Marvin Cheng, Dan B. Goldman, Steven M. Seitz, and Dieter Fox. Interactive 3d modeling of indoor environments with a consumer depth camera. In *Proc. of the 13th Intl. conf. on Ubiquitous computing*, UbiComp '11, pages 75–84, New York, NY, USA, 2011. ACM.

[51] Christopher B Eiben, Justin B Siegel, Jacob B Bale, Seth Cooper, Firas Khatib, Betty W Shen, Foldit Players, Barry L Stoddard, Zoran Popovic, and David Baker. Increased diels-alderase activity through backbone remodeling guided by foldit players. *Nature biotechnology*, 30(2):190–192, 2012.

[52] Paul Ekman. Basic emotions. *Handbook of cognition and emotion*, 98:45–60, 1999.

[53] Paul Ekman and Wallace V Friesen. *Facial action coding system*. Consulting Psychologists Press, Stanford University, Palo Alto, 1977.

[54] Katharina Emmerich and Maic Masuch. Helping friends or fighting foes: The influence of collaboration and competition on player experience. In *FDG*, pages 150–157, 2013.

[55] Alan Eustace. A fall spring-clean. `http://googleblog.blogspot.com/2011/09/fall-spring-clean.html`, 2011.

[56] Claire Lisa Evans. *Broad band: the untold story of the women who made the Internet*. Penguin, 2018.

[57] Facepunch Studios. Garry's mod. `http://garrysmod.com/`.

[58] Li Fei-Fei, R Fergus, and P Perona. Learning Generative Visual Models from Few Training Examples: An Incremental Bayesian Approach Tested on 101 Object Categories. In *Computer Vision and Pattern Recognition Workshop, 2004. CVPRW '04. Conference on*, page 178, 2004.

[59] Li Fei-Fei, Rob Fergus, and Pietro Perona. Learning generative visual models from few training examples: An incremental bayesian approach tested on 101 object categories. *Computer vision and Image understanding*, 106(1):59–70, 2007.

[60] Jan-Michael Frahm, Pierre Fite-Georgel, David Gallup, Tim Johnson, Rahul Raguram, Changchang Wu, Yi-Hung Jen, Enrique Dunn, Brian Clipp, Svetlana Lazebnik, et al. Building rome on a cloudless day. In *European Conference on Computer Vision*, pages 368–381. Springer, 2010.

[61] Tracy Fullerton, Chris Swain, and Steven Hoffman. *Game Design Workshop*. Designing, Prototyping, & Playtesting Games. CRC Press, January 2004.

[62] Thomas Funkhouser, Michael Kazhdan, Philip Shilane, Patrick Min, William Kiefer, Ayellet Tal, Szymon Rusinkiewicz, and David Dobkin. Modeling by example. In *ACM SIGGRAPH 2004 Papers*, SIGGRAPH '04, pages 652–663, New York, NY, USA, 2004. ACM.

[63] Yasutaka Furukawa, Brian Curless, Steven M Seitz, and Richard Szeliski. Manhattan-world stereo. In *Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on*, pages 1422–1429. IEEE, 2009.

[64] Yasutaka Furukawa, Brian Curless, Steven M. Seitz, and Richard Szeliski. Towards internet-scale multi-view stereo. In *CVPR*, 2010.

[65] Yasutaka Furukawa and Jean Ponce. Patch-based multi-view stereo software. http://grail.cs.washington.edu/software/pmvs.

[66] Yasutaka Furukawa and Jean Ponce. Accurate, dense, and robust multiview stereopsis. *IEEE transactions on pattern analysis and machine intelligence*, 32(8):1362–1376, 2010.

[67] Timnit Gebru, Jamie Morgenstern, Briana Vecchione, Jennifer Wortman Vaughan, Hanna Wallach, Hal Daumeé III, and Kate Crawford. Datasheets for datasets. *arXiv preprint arXiv:1803.09010*, 2018.

[68] Dave Gershgorn. The data that transformed ai research–and possibly the world. *Quartz. Quartz. July*, 26:2017, 2017.

[69] Michael F Goodchild and Linna Li. Assuring the quality of volunteered geographic information. *Spatial statistics*, 1:110–120, 2012.

[70] Ian J Goodfellow, Dumitru Erhan, Pierre Luc Carrier, Aaron Courville, Mehdi Mirza, Ben Hamner, Will Cukierski, Yichuan Tang, David Thaler, Dong-Hyun Lee, Yingbo Zhou, Chetan Ramaiah, Fangxiang Feng, Ruifan Li, Xiaojie Wang, Dimitris Athanasakis, John Shawe-Taylor, Maxim Milakov, John Park, Radu Ionescu, Marius Popescu, Cristian Grozea, James Bergstra, Jingjing Xie, Lukasz Romaszko, Bing Xu, Zhang Chuang, and Yoshua Bengio. Challenges in representation learning: A report on three machine learning contests. *Neural Networks*, 2014.

[71] Gregory Griffin, Alex Holub, and Pietro Perona. Caltech-256 object category dataset, 2007.

[72] R Gross, I Matthews, J Cohn, T Kanade, and S Baker. Multi-PIE. In *Automatic Face & Gesture Recognition, 2008. FG '08. 8th IEEE International Conference on*, pages 1–8, 2008.

[73] Ralph Gross, Iain Matthews, Jeffrey Cohn, Takeo Kanade, and Simon Baker. Multi-pie. *Image and Vision Computing*, 28(5):807–813, 2010.

[74] Udutu Guru. Gamification: Engaging learners with game techniques. `http://blog.udutu.com/blog/bid/302280/Gamification-Engaging-Learners-with-Game-Techniques`, 2013.

[75] Mordechai Haklay and Patrick Weber. Openstreetmap: User-generated street maps. *IEEE Pervasive Computing*, 7(4):12–18, 2008.

[76] Peter Henry, Michael Krainin, Evan Herbst, Xiaofeng Ren, and Dieter Fox. Rgbd mapping: Using depth cameras for dense 3d modeling of indoor environments. In *In RGB-D: Advanced Reasoning with Depth Cameras Workshop in conjunction with RSS*, 2010.

[77] Marie Hicks. *Programmed inequality: How Britain discarded women technologists and lost its edge in computing*. MIT Press, 2017.

[78] Chien-Ju Ho, Tao-Hsuan Chang, Jong-Chuan Lee, Jane Yung-jen Hsu, and Kuan-Ta Chen. Kisskissban: a competitive human computation game for image annotation. *ACM SIGKDD Explorations Newsletter*, 12(1):21–24, 2010.

[79] Hugues Hoppe, Tony DeRose, Tom Duchamp, John McDonald, and Werner Stuetzle. Surface reconstruction from unorganized points. In *Proc. of the 19th annual conf. on Computer graphics and interactive techniques*, SIGGRAPH '92, pages 71–78, New York, NY, USA, 1992. ACM.

[80] J Howe. The rise of crowdsourcing. *Wired magazine*, 2006.

[81] Gary B Huang, Marwan Mattar, Tamara Berg, and Eric Learned-Miller. Labeled faces in the wild: A database forstudying face recognition in unconstrained environments. In *Workshop on faces in'Real-Life'Images: detection, alignment, and recognition*, 2008.

[82] Takeo Igarashi, Satoshi Matsuoka, and Hidehiko Tanaka. Teddy: a sketching interface for 3d freeform design. In *ACM SIGGRAPH 2006 Courses*, SIGGRAPH '06, New York, NY, USA, 2006. ACM.

[83] Megvii Inc. Face++ research toolkit. www.faceplusplus.com, December 2013.

[84] Arnold Irschara, Christopher Zach, and Horst Bischof. Towards wiki-based dense city modeling. In *2007 ieee 11th international conference on computer vision*, pages 1–8. IEEE, 2007.

[85] Shahram Izadi, David Kim, Otmar Hilliges, David Molyneaux, Richard Newcombe, Pushmeet Kohli, Jamie Shotton, Steve Hodges, Dustin Freeman, Andrew Davison, et al. Kinectfusion: real-time 3d reconstruction and interaction using a moving depth camera. In *Proceedings of the 24th annual ACM symposium on User interface software and technology*, pages 559–568. ACM, 2011.

[86] Jesper Juul and Marleigh Norton. Easy to use and incredibly difficult: on the mythical border between interface and gameplay. In *FDG '09: Proceedings of the 4th International Conference on Foundations of Digital Games*, pages 107–112, New York, NY, USA, 2009. ACM.

[87] Takeo Kanade, Jeffrey F Cohn, and Yingli Tian. Comprehensive database for facial expression analysis. In *Automatic Face and Gesture Recognition, 2000. Proceedings. Fourth IEEE International Conference on*, pages 46–53. IEEE, 2000.

[88] Andrej Kaparthy. t-sne visualization of cnn codes. `https://cs.stanford.edu/people/karpathy/cnnembed/`, 2009. Accessed: 2019-05-21.

[89] Alexander Kawrykow, Gary Roumanis, Alfred Kam, Daniel Kwak, Clarence Leung, Chu Wu, Eleyine Zarour, Luis Sarmenta, Mathieu Blanchette, Jérôme Waldispühl, et al. Phylo: a citizen science approach for improving multiple sequence alignment. *PloS one*, 7(3):e31362, 2012.

[90] Michael Kazhdan, Matthew Bolitho, and Hugues Hoppe. Poisson surface reconstruction. In *Proc. of the fourth Eurographics symposium on Geometry processing*, SGP '06, pages 61–70, Aire-la-Ville, Switzerland, Switzerland, 2006. Eurographics Association.

[91] Tom Kelly and Peter Wonka. Interactive architectural modeling with procedural extrusions. *ACM Trans. Graph.*, 30:14:1–14:15, April 2011.

[92] I Kemelmacher-Shlizerman and S M Seitz. Face reconstruction in the wild. In *Computer Vision (ICCV), 2011 IEEE International Conference on*, pages 1746–1753, 2011.

[93] Ira Kemelmacher-Shlizerman and Steven M Seitz. Collection flow. In *Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conference on*, pages 1792–1799. IEEE, 2012.

[94] Ira Kemelmacher-Shlizerman, Supasorn Suwajanakorn, and Steven M Seitz. Illumination-aware age progression. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3334–3341, 2014.

[95] Lyndon Kennedy, Mor Naaman, Shane Ahern, Rahul Nair, and Tye Rattenbury. How Flickr helps us make sense of the world: Context and content in community-contributed media collections. In *Proc. Int. Conf. on Multimedia*, 2007.

[96] Os Keyes. The misgendering machines: Trans/hci implications of automatic gender recognition. *Proceedings of the ACM on Human-Computer Interaction*, 2(CSCW):88, 2018.

[97] Os Keyes, Josephine Hoy, and Margaret Drouhard. Human-computer insurrection: Notes on an anarchist hci. In *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems*, page 339. ACM, 2019.

[98] Firas Khatib, Seth Cooper, Michael D Tyka, Kefan Xu, Ilya Makedon, Zoran Popović, David Baker, and Foldit Players. Algorithm discovery by protein folding game players. *Proceedings of the National Academy of Sciences*, 108(47):18949–18953, November 2011.

[99] Firas Khatib, Frank DiMaio, Seth Cooper, Maciej Kazmierczyk, Miroslaw Gilski, Szymon Krzywda, Helena Zabranska, Iva Pichova, James Thompson, Zoran Popović, et al. Crystal structure of a monomeric retroviral protease solved by protein folding game players. *Nature structural & molecular biology*, 18(10):1175–1177, 2011.

[100] Jinseop S Kim, Matthew J Greene, Aleksandar Zlateski, Kisuk Lee, Mark Richardson, Srinivas C Turaga, Michael Purcaro, Matthew Balkam, Amy Robinson, Bardia F Behabadi, et al. Space–time wiring specificity supports direction selectivity in the retina. *Nature*, 509(7500):331, 2014.

[101] Aniket Kittur, Jeffrey V Nickerson, Michael Bernstein, Elizabeth Gerber, Aaron Shaw, John Zimmerman, Matt Lease, and John Horton. The future of crowd work. In *CSCW '13: Proceedings of the 2013 conference on Computer supported cooperative work*. ACM Request Permissions, February 2013.

[102] Aniket Kittur, Boris Smus, Susheel Khamkar, and Robert E Kraut. CrowdForge: crowdsourcing complex work. In *UIST '11*. ACM Request Permissions, October 2011.

[103] Leif Kobbelt and Mario Botsch. A survey of point-based techniques in computer graphics. *Computers & Graphics*, 28(6):801–814, 2004.

[104] Raph Koster and Will Wright. *A Theory of Fun for Game Design*. Paraglyph Press, 2004.

[105] Adarsh Kowdle, Yao-Jen Chang, Andrew Gallagher, and Tsuhan Chen. Active learning for piecewise planar 3d reconstruction. In *CVPR*. IEEE, 2011.

[106] Markus Krause, Aneta Takhtamysheva, Marion Wittstock, and Rainer Malaka. Frontiers of a paradigm: Exploring human computation with digital games. In *Proceedings of the ACM SIGKDD Workshop on Human Computation*, HCOMP '10, pages 22–25, New York, NY, USA, 2010. ACM.

[107] Travis Kriplean, Jonathan Morgan, Deen Freelon, Alan Borning, and Lance Bennett. Supporting reflective public thought with considerit. In *Proceedings of the ACM 2012 conference on Computer Supported Cooperative Work*, pages 265–274. ACM, 2012.

[108] N Kumar, A C Berg, P N Belhumeur, and S K Nayar. Attribute and simile classifiers for face verification. In *Computer Vision, 2009 IEEE 12th International Conference on*, pages 365–372, 2009.

[109] N Kumar, A C Berg, P N Belhumeur, and S K Nayar. Describable Visual Attributes for Face Verification and Image Search. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 33(10):1962–1977, 2011.

[110] Neeraj Kumar, Peter Belhumeur, and Shree Nayar. Facetracer: A search engine for large collections of images with faces. In *Computer Vision–ECCV 2008*, pages 340–353. Springer, 2008.

[111] Neeraj Kumar, Peter N Belhumeur, Arijit Biswas, David W Jacobs, W John Kress, Ida C Lopez, and João VB Soares. Leafsnap: A computer vision system for automatic plant species identification. In *European Conference on Computer Vision*, pages 502–516. Springer, 2012.

[112] Neeraj Kumar, Alexander C Berg, Peter N Belhumeur, and Shree K Nayar. Attribute and simile classifiers for face verification. In *Computer Vision, 2009 IEEE 12th International Conference on*, pages 365–372. IEEE, 2009.

[113] Alina Kuznetsova, Hassan Rom, Neil Alldrin, Jasper Uijlings, Ivan Krasin, Jordi Pont-Tuset, Shahab Kamali, Stefan Popov, Matteo Malloci, Tom Duerig, et al. The open images dataset v4: Unified image classification, object detection, and visual relationship detection at scale. *arXiv preprint arXiv:1811.00982*, 2018.

[114] Edith LM Law, Luis Von Ahn, Roger B Dannenberg, and Mike Crawford. Tagatune: A game for music and sound annotation. In *ISMIR*, volume 3, page 2, 2007.

[115] Jeehyung Lee, Wipapat Kladwang, Minjae Lee, Daniel Cantu, Martin Azizyan, Hanjoo Kim, Alex Limpaecher, Sungroh Yoon, Adrien Treuille, and Rhiju Das. Rna design rules from a massive open laboratory. *Proceedings of the National Academy of Sciences*, 111(6):2122–2127, 2014.

[116] Bol Leuf and Ward Cunningham. *The Wiki way: quick collaboration on the Web*, volume 9. Addison-Wesley Boston, 2001.

[117] Marc Levoy, Kari Pulli, Brian Curless, Szymon Rusinkiewicz, David Koller, Lucas Pereira, Matt Ginzton, Sean Anderson, James Davis, Jeremy Ginsberg, Jonathan Shade, and Duane Fulk. The digital michelangelo project: 3d scanning of large statues. In *Proc. of the 27th annual conf. on Computer graphics and interactive techniques*, SIGGRAPH '00, pages 131–144, New York, NY, USA, 2000. ACM Press/Addison-Wesley Publishing Co.

[118] Xiaobai Li, Tomas Pfister, Xiaohua Huang, Guoying Zhao, and Matti Pietikainen. A spontaneous micro-expression database: Inducement, collection and baseline. In *Automatic Face and Gesture Recognition (FG), 2013 10th IEEE International Conference and Workshops on*, pages 1–6. IEEE, 2013.

[119] Yangyan Li, Xiaokun Wu, Yiorgos Chrysathou, Andrei Sharf, Daniel Cohen-Or, and Niloy J. Mitra. Globfit: consistently fitting primitives by discovering global relations. In *ACM SIGGRAPH 2011 papers*, SIGGRAPH '11, pages 52:1–52:12, New York, NY, USA, 2011. ACM.

[120] Mark Limber. Introducing google building maker. `https://googleblog.blogspot.com/2009/10/introducing-google-building-maker.html`, 2009. Accessed: 2019-04-19.

[121] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. Microsoft coco: Common objects in context. In *European conference on computer vision*, pages 740–755. Springer, 2014.

[122] Greg Little, Lydia B Chilton, Max Goldman, and Robert C Miller. TurKit: human computation algorithms on mechanical turk. In *UIST '10*. ACM Request Permissions, October 2010.

[123] Yotam Livny, Feilong Yan, Matt Olson, Baoquan Chen, Hao Zhang, and Jihad El-Sana. Automatic reconstruction of tree skeletal structures from point clouds. In *ACM SIGGRAPH Asia 2010 papers*, SIGGRAPH ASIA '10, pages 151:1–151:8, New York, NY, USA, 2010. ACM.

[124] Patrick Lucey, Jeffrey F Cohn, Takeo Kanade, Jason Saragih, Zara Ambadar, and Iain Matthews. The extended cohn-kanade dataset (ck+): A complete dataset for action unit and emotion-specified expression. In *Computer Vision and Pattern Recognition Workshops (CVPRW), 2010 IEEE Computer Society Conference on*, pages 94–101. IEEE, 2010.

[125] Laurens van der Maaten and Geoffrey Hinton. Visualizing data using t-sne. *Journal of machine learning research*, 9(Nov):2579–2605, 2008. Olivetti Face t-SNE URL: `https://lvdmaaten.github.io/tsne/examples/olivetti_tsne.jpg`.

[126] Douglas MacMillan and Brad Stone. Zynga's quest for big-spending whales. `http://www.businessweek.com/magazine/zyngas-quest-for-bigspending-whales-07072011.html`, 2011.

[127] Ricardo Martin-Brualla, David Gallup, and Steven M Seitz. Time-lapse mining from internet photos. *ACM Transactions on Graphics (TOG)*, 34(4):62, 2015.

[128] Sebastian Matyas, Christian Matyas, Christoph Schlieder, Peter Kiefer, Hiroko Mitarai, and Maiko Kamata. Designing location-based mobile games with a purpose: collecting geospatial data with cityexplorer. In *Proceedings of the 2008 International Conference on Advances in Computer Entertainment Technology*, pages 244–247. ACM, 2008.

[129] Kevin Matzen, Kavita Bala, and Noah Snavely. Streetstyle: Exploring world-wide clothing styles from millions of photos. *arXiv preprint arXiv:1706.01869*, 2017.

[130] Steve McConnell. Productivity variations among software developers and teams: The origin of 10x. `http://www.construx.com/10x_Software_Development/Productivity_Variations_Among_Software_Developers_and_Teams__The_Origin_of_10x/`, 2008.

[131] D McDuff, R El Kaliouby, and T Senechal. Automatic Measurement of Ad Preferences from Facial Responses Gathered Over the Internet. *Image and Vision ...*, 2014.

[132] Daniel McDuff, Rana el Kaliouby, and Rosalind Picard. Crowdsourced data collection of facial responses. In *ICMI '11: Proceedings of the 13th international conference on multimodal interfaces*. ACM Request Permissions, November 2011.

[133] Daniel McDuff, Rana El Kaliouby, Thibaud Senechal, May Amr, Jeffrey F Cohn, and Rosalind Picard. Affectiva-mit facial expression dataset (am-fed): Naturalistic and spontaneous facial expressions collected" in-the-wild". In *Computer Vision and Pattern Recognition Workshops (CVPRW), 2013 IEEE Conference on*, pages 881–888. IEEE, 2013.

[134] Chris Milk and A Koblin. The johnny cash project. `http://www.thejohnnycashproject.com/`, 2010. Additional URL: `https://docubase.mit.edu/project/the-johnny-cash-project/`.

[135] George A Miller. Wordnet: a lexical database for english. *Communications of the ACM*, 38(11):39–41, 1995.

[136] Vikram Mohanty, David Thames, Sneha Mehta, and Kurt Luther. Photo sleuth: Combining human expertise and face recognition to identify historical portraits. In *Proceedings of the 24th International Conference on Intelligent User Interfaces*, pages 547–557. ACM, 2019.

[137] Janet Horowitz Murray. *Hamlet on the holodeck: The future of narrative in cyberspace*. Simon and Schuster, 1997.

[138] Liangliang Nan, Andrei Sharf, Hao Zhang, Daniel Cohen-Or, and Baoquan Chen. Smartboxes for interactive urban reconstruction. In *ACM Transactions on Graphics (TOG)*, volume 29, page 93. ACM, 2010.

[139] Mark J Nelson. Soviet and American precursors to the gamification of work. In *MindTrek '12: Proceeding of the 16th International Academic MindTrek Conference*. ACM Request Permissions, October 2012.

[140] Byong Mok Oh, Max Chen, Julie Dorsey, and Frédo Durand. Image-based modeling and photo editing. In *Proc. of the 28th annual conf. on Computer graphics and interactive techniques*, SIGGRAPH '01, pages 433–442, New York, NY, USA, 2001. ACM.

[141] Sangmin Oh, Anthony Hoogs, Amitha Perera, Naresh Cuntoor, Chia-Chih Chen, Jong Taek Lee, Saurajit Mukherjee, JK Aggarwal, Hyungtae Lee, Larry Davis, et al. A large-scale benchmark dataset for event recognition in surveillance video. In *CVPR 2011*, pages 3153–3160. IEEE, 2011.

[142] Luke Olsen, Faramarz F. Samavati, Mario Costa Sousa, and Joaquim A. Jorge. Technical section: Sketch-based modeling: A survey. *Comput. Graph.*, 33:85–103, February 2009.

[143] Maja Pantic, Michel Valstar, Ron Rademaker, and Ludo Maat. Web-based database for facial expression analysis. In *Multimedia and Expo, 2005. ICME 2005. IEEE International Conference on*, pages 5–pp. IEEE, 2005.

[144] Dim P Papadopoulos, Jasper RR Uijlings, Frank Keller, and Vittorio Ferrari. We don't need no bounding-boxes: Training object class detectors using only human verification. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 854–863, 2016.

[145] Kayur Patel, Naomi Bancroft, Steven M Drucker, James Fogarty, Andrew J Ko, and James Landay. Gestalt: integrated support for implementation and analysis in machine learning. In *UIST '10: Proceedings of the 23nd annual ACM symposium on User interface software and technology*. ACM Request Permissions, October 2010.

[146] Genevieve Patterson and James Hays. Sun attribute database: Discovering, annotating, and recognizing scene attributes. In *2012 IEEE Conference on Computer Vision and Pattern Recognition*, pages 2751–2758. IEEE, 2012.

[147] Mark Pauly, Richard Keiser, Leif P. Kobbelt, and Markus Gross. Shape modeling with point-sampled geometry. In *ACM SIGGRAPH 2003 Papers*, SIGGRAPH '03, pages 641–650, New York, NY, USA, 2003. ACM.

[148] Markus Persson. Minecraft statistics. `https://minecraft.net/stats`. Accessed: 2015-04-20.

[149] Daniel H Pink. *Drive*. The Surprising Truth about what Motivates Us. Penguin, 2009.

[150] Bentley systems pointools plug-in for sketchup. `http://pointools.com/pdf/Pointools-Plug-in-for-SketchUp.pdf`.

[151] Nathan Prestopnik and Kevin Crowston. Exploring collective intelligence games with design science: A citizen science design case. In *ACM Group Conference*, 2012.

[152] Till Quack, Bastian Leibe, and Luc Van Gool. World-scale mining of objects and events from community photo collections. In *Proc. Int. Conf. on Content-based Image and Video Retrieval*, pages 47–56, 2008.

[153] M Jordan Raddick, Georgia Bracey, Pamela L Gay, Chris J Lintott, Phil Murray, Kevin Schawinski, Alexander S Szalay, and Jan Vandenberg. Galaxy zoo: Exploring the motivations of citizen science volunteers. *arXiv preprint arXiv:0909.2925*, 2009.

[154] Deva Ramanan and Xiangxin Zhu. Face detection, pose estimation, and landmark localization in the wild. In *2012 IEEE conference on computer vision and pattern recognition*, pages 2879–2886. IEEE, 2012.

[155] Ronald Reisman and Rana el Kaliouby. Facial expression affective state recognition for air traffic control automation concept exploration. *SIGGRAPH '07: SIGGRAPH 2007 posters*, August 2007.

[156] Mitchel Resnick, John Maloney, Andrés Monroy-Hernández, Natalie Rusk, Evelyn Eastmond, Karen Brennan, Amon Millner, Eric Rosenbaum, Jay S Silver, Brian Silverman, et al. Scratch: Programming for all. *Commun. Acm*, 52(11):60–67, 2009.

[157] Andrew Rollings and Dave Morris. *Game Architecture and Design with Cdrom*. Coriolis Group Books, 1999.

[158] Ricarose Roque, Natalie Rusk, and Mitchel Resnick. Supporting diverse and creative collaboration in the scratch online community. In *Mass collaboration and education*, pages 241–256. Springer, 2016.

[159] Eleanor Rosch and Barbara Bloom Lloyd. Cognition and categorization, 1978.

[160] Bryan C Russell, Ricardo Martin-Brualla, Daniel J Butler, Steven M Seitz, and Luke Zettlemoyer. 3d wikipedia: Using online text to automatically label and navigate reconstructed geometry. *ACM Transactions on Graphics (TOG)*, 32(6):193, 2013.

[161] Bryan C Russell, Antonio Torralba, Kevin P Murphy, and William T Freeman. Labelme: a database and web-based tool for image annotation. *International journal of computer vision*, 77(1-3):157–173, 2008.

[162] Marta Sabou, Kalina Bontcheva, Arno Scharl, and Michael Föls. Games with a Purpose or Mechanised Labour?: A Comparative Study. In *i-Know '13: Proceedings of the 13th International Conference on Knowledge Management and Knowledge Technologies*. ACM Request Permissions, September 2013.

[163] H. Sackman, W. J. Erikson, and E. E. Grant. Exploratory experimental studies comparing online and offline programming performance. *Commun. ACM*, 11(1):3–11, January 1968.

[164] Manaswi Saha, Michael Saugstad, Hanuma Teja Maddali, Aileen Zeng, Ryan Holland, Steven Bower, Aditya Dash, Sage Chen, Anthony Li, Kotaro Hara, and Jon Froehlich. Project sidewalk: A web-based crowdsourcing tool for collecting sidewalk accessibility data at scale. In *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems*, CHI '19, pages 62:1–62:14, New York, NY, USA, 2019. ACM.

[165] Niloufar Salehi, Lilly C Irani, Michael S Bernstein, Ali Alkhatib, Eva Ogbe, Kristy Milland, et al. We are dynamo: Overcoming stalling and friction in collective action for crowd workers. In *Proceedings of the 33rd annual ACM conference on human factors in computing systems*, pages 1621–1630. ACM, 2015.

[166] Neil Savage and Neil Savage. Gaining wisdom from crowds. *Communications of the ACM*, 55(3):13, March 2012.

[167] D Scharstein and R Szeliski. A taxonomy and evaluation of dense two-frame stereo correspondence algorithms. *International journal of computer vision*, 2002.

[168] Ryan Schmidt, Azam Khan, Karan Singh, and Gord Kurtenbach. Analytic drawing of 3d scaffolds. In *ACM SIGGRAPH Asia 2009 papers*, SIGGRAPH Asia '09, pages 149:1–149:10, New York, NY, USA, 2009. ACM.

[169] Ruwen Schnabel, Roland Wahl, and Reinhard Klein. Efficient ransac for point-cloud shape detection. *Computer Graphics Forum*, 26(2):214–226, June 2007.

[170] Qi Shan, R. Adams, B. Curless, Y. Furukawa, and S.M. Seitz. The visual turing test for scene reconstruction. In *3DV-Conference, 2013 International Conference on*, pages 25–32, June 2013.

[171] Ian Simon, Noah Snavely, and Steven M Seitz. Scene summarization for online image collections. In *2007 IEEE 11th International Conference on Computer Vision*, pages 1–8. IEEE, 2007.

[172] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.

[173] Robert Simpson, Kevin R Page, and David De Roure. Zooniverse: observing the world's largest citizen science platform. In *Proceedings of the 23rd international conference on world wide web*, pages 1049–1054. ACM, 2014.

[174] Sudipta N. Sinha, Drew Steedly, and Richard Szeliski. Piecewise planar stereo for image-based rendering. In *ICCV*, pages 1881–1888, 2009.

[175] Sudipta N. Sinha, Drew Steedly, Richard Szeliski, Maneesh Agrawala, and Marc Pollefeys. Interactive 3d architectural modeling from unordered photo collections. In *ACM SIGGRAPH Asia 2008 papers*, SIGGRAPH Asia '08, pages 159:1–159:10, New York, NY, USA, 2008. ACM.

[176] Kristin Siu and Mark O Riedl. Reward systems in human computation games. In *Proceedings of the 2016 Annual Symposium on Computer-Human Interaction in Play*, pages 266–275. ACM, 2016.

[177] Kristin Siu, Alexander Zook, and Mark O Riedl. Collaboration versus competition: Design and evaluation of mechanics for games with a purpose. In *FDG*, 2014.

[178] Kristin Siu, Alexander Zook, and Mark O Riedl. A framework for exploring and evaluating mechanics in human computation games. In *Proceedings of the 12th International Conference on the Foundations of Digital Games*, page 38. ACM, 2017.

[179] Noah Snavely. Bundler: Structure from motion (sfm) for unordered image collections. *http://phototour. cs. washington. edu/bundler/*, 2008.

[180] Noah Snavely, Rahul Garg, Steven M Seitz, and Richard Szeliski. Finding paths through the world's photos. In *ACM Transactions on Graphics (TOG)*, volume 27, page 15. ACM, 2008.

[181] Noah Snavely, Steven M. Seitz, and Richard Szeliski. Photo tourism: exploring photo collections in 3d. In *ACM SIGGRAPH 2006 Papers*, SIGGRAPH '06, pages 835–846, New York, NY, USA, 2006. ACM.

[182] Ian Spiro. Motion chain: a webcam game for crowdsourcing gesture collection. In *CHI'12 Extended Abstracts on Human Factors in Computing Systems*, pages 1345–1350. ACM, 2012.

[183] Kate Starbird and Jeannie Stamberger. Tweak the tweet: Leveraging microblogging proliferation with a prescriptive syntax to support citizen reporting. In *Proceedings of the 7th International ISCRAM Conference*, volume 1, pages 1–5. Information Systems for Crisis Response and Management Seattle, WA, 2010.

[184] Tobias Sturn, Michael Wimmer, Peter Purgathofer, and Steffen Fritz. Landspotting - games for improving global land cover. In *Proceedings of Foundations of Digital Games Conference 2013 (FDG 2013)*, pages 117–125, May 2013.

[185] Supasorn Suwajanakorn, Ira Kemelmacher-Shlizerman, and Steven M Seitz. Total moving face reconstruction. In *European conference on computer vision*, pages 796–812. Springer, 2014.

[186] Supasorn Suwajanakorn, Steven M Seitz, and Ira Kemelmacher-Shlizerman. What makes tom hanks look like tom hanks. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 3952–3960, 2015.

[187] Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. Going deeper with convolutions. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1–9, 2015.

[188] Yaniv Taigman, Ming Yang, Marc'Aurelio Ranzato, and Lior Wolf. Deepface: Closing the gap to human-level performance in face verification. In *Computer Vision and Pattern Recognition (CVPR), 2014 IEEE Conference on*, pages 1701–1708. IEEE, 2014.

[189] Jerry O. Talton, Daniel Gibson, Lingfeng Yang, Pat Hanrahan, and Vladlen Koltun. Exploratory modeling with collaborative design spaces. In *ACM SIGGRAPH Asia 2009 papers*, SIGGRAPH Asia '09, pages 167:1–167:10, New York, NY, USA, 2009. ACM.

[190] Chek Tien Tan, Hemanta Sapkota, Daniel Rosser, and Yusuf Pisan. A game to crowdsource data for affective computing. *Proceedings of Foundations of Digital Games*, page 11, 2014.

[191] Kathleen Tuite. Photocity archive. `http://photocitygame.com`. Accessed: 2015-04-20.

[192] Kathleen Tuite. GWAPs: Games With a Problem. In *FDG '14: Proceedings of the International Conference on the Foundations of Digital Games*. ACM, April 2014.

[193] Kathleen Tuite, Rahul Banerjee, Noah Snavely, Jovan Popovic, and Zoran Popovic. Pointcraft: Harnessing players' fps skills to interactively trace point clouds in 3d. In *FDG '15: Proceedings of the International Conference on the Foundations of Digital Games*, 2015.

[194] Kathleen Tuite and Ira Kemelmacher. The meme quiz: A facial expression game combining human agency and machine involvement. In *FDG '15: Proceedings of the International Conference on the Foundations of Digital Games*, 2015.

[195] Kathleen Tuite, Timothy Pavlik, Sandra B. Fan, Tyler Robison, Alexander Jaffe, Yun-En Liu, Erik Andersen, and Steven Tanimoto. Picard: A creative and social online flashcard learning game. In *Proceedings of the International Conference on the Foundations of Digital Games*, FDG '12, pages 231–234, New York, NY, USA, 2012. ACM.

[196] Kathleen Tuite and Adam M Smith. Emergent remix culture in an anonymous collaborative art system. In *Workshop on Human Computation in Digital Entertainment at AIIDE 2012*, 2012.

[197] Kathleen Tuite, Noah Snavely, Dun-Yu Hsiao, Adam M Smith, and Zoran Popović. Reconstructing the world in 3d: bringing games with a purpose outdoors. In *Proceedings of the Fifth International Conference on the Foundations of Digital Games*, pages 232–239. ACM, 2010.

[198] Kathleen Tuite, Noah Snavely, Dun-yu Hsiao, Nadine Tabing, and Zoran Popovic. Photocity: training experts at large-scale image acquisition through a competitive game. In *Proc. of the 2011 annual conf. on Human factors in computing systems*, CHI '11, pages 1383–1392, New York, NY, USA, 2011. ACM.

[199] Robert J Vallerand. Toward A Hierarchical Model of Intrinsic and Extrinsic Motivation. In *Advances in experimental social psychology*, pages 271–360. Elsevier, 1997.

[200] Anton van den Hengel, Anthony Dick, Thorsten Thormählen, Ben Ward, and Philip H. S. Torr. Videotrace: rapid interactive scene modelling from video. In *ACM SIGGRAPH 2007 papers*, SIGGRAPH '07, New York, NY, USA, 2007. ACM.

[201] Grant Van Horn, Oisin Mac Aodha, Yang Song, Yin Cui, Chen Sun, Alex Shepard, Hartwig Adam, Pietro Perona, and Serge Belongie. The inaturalist species classification and detection dataset. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 8769–8778, 2018.

[202] Nancy A. Van House. Flickr and public image-sharing: distant closeness and photo exhibition. In *CHI '07: CHI '07 extended abstracts on Human factors in computing systems*, pages 2717–2722, New York, NY, USA, 2007. ACM.

[203] Andreas Veit, Balazs Kovacs, Sean Bell, Julian McAuley, Kavita Bala, and Serge Belongie. Learning visual clothing style with heterogeneous dyadic co-occurrences. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 4642–4650, 2015.

[204] Raviteja Vemulapalli and Aseem Agarwala. A compact embedding for facial expression similarity. *CoRR*, abs/1811.11283, 2018.

[205] Paul Viola and Michael J Jones. Robust real-time face detection. *International journal of computer vision*, 57(2):137–154, 2004.

[206] Luis Von Ahn. Duolingo: learn a language for free while helping to translate the web. In *Proceedings of the 2013 international conference on Intelligent user interfaces*, pages 1–2. ACM, 2013.

[207] Luis Von Ahn and Laura Dabbish. Labeling images with a computer game. In *Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 319–326. ACM, 2004.

[208] Luis von Ahn and Laura Dabbish. Designing games with a purpose. *Commun. ACM*, 51(8):58–67, August 2008.

[209] Luis Von Ahn, Mihir Kedia, and Manuel Blum. Verbosity: a game for collecting common-sense facts. In *Proceedings of the SIGCHI conference on Human Factors in computing systems*, pages 75–78. ACM, 2006.

[210] Luis Von Ahn, Ruoran Liu, and Manuel Blum. Peekaboom: a game for locating objects in images. In *Proceedings of the SIGCHI conference on Human Factors in computing systems*, pages 55–64. ACM, 2006.

[211] Carl Vondrick, Donald Patterson, and Deva Ramanan. Efficiently scaling up crowdsourced video annotation. *International Journal of Computer Vision*, 101(1):184–204, 2013.

[212] Carl Vondrick and Deva Ramanan. Video annotation and tracking with active learning. In *Advances in Neural Information Processing Systems*, pages 28–36, 2011.

[213] Noah Wardrip-Fruin. Three play effects–eliza, tale-spin, and sim city. *Digital Humanities*, pages 1–2, 2007.

[214] Noah Wardrip-Fruin. *Expressive Processing: Digital fictions, computer games, and software studies*. MIT press, 2009.

[215] T. Weyrich, M. Pauly, R. Keiser, S. Heinzle, S. Scandella, and M. Gross. Post-processing of scanned 3d surface data. In *Proceedings of the First Eurographics conference on Point-Based Graphics*, SPBG'04, pages 85–94, Aire-la-Ville, Switzerland, Switzerland, 2004. Eurographics Association.

[216] Eric Whitacre. The virtual choir. *Eric Whitacre Website. Accessed February*, 15, 2014.

[217] Chris Wood, Brian Sullivan, Marshall Iliff, Daniel Fink, and Steve Kelling. ebird: engaging birders in science and conservation. *PLoS biology*, 9(12):e1001220, 2011.

[218] Xuehan-Xiong and Fernando De la Torre. Supervised descent method and its application to face alignment. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2013.

[219] Wen-Jing Yan, Qi Wu, Yong-Jin Liu, Su-Jing Wang, and Xiaolan Fu. Casme database: a dataset of spontaneous micro-expressions collected from neutralized faces. In *Automatic Face and Gesture Recognition (FG), 2013 10th IEEE International Conference and Workshops on*, pages 1–7. IEEE, 2013.

[220] Nick Yee, Nicolas Ducheneaut, and Les Nelson. Online gaming motivations scale: development and validation. In *CHI '12: Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. ACM Request Permissions, May 2012.

[221] Stefanos Zafeiriou, Gary A Atkinson, Mark F Hansen, William AP Smith, Vasileios Argyriou, Maria Petrou, Melvyn L Smith, and Lyndon N Smith. Face recognition and verification using photometric stereo: The photoface database and a comprehensive evaluation. *IEEE transactions on information forensics and security*, 8(1):121–135, 2013.

[222] José P Zagal, Staffan Björk, and Chris Lewis. Dark patterns in the design of games. In *Foundations of Digital Games 2013*, 2013.

[223] Robert C. Zeleznik, Kenneth P. Herndon, and John F. Hughes. Sketch: An interface for sketching 3d scenes. In *Proceedings of the 23rd Annual Conference on Computer Graphics and Interactive Techniques*, SIGGRAPH '96, pages 163–170, New York, NY, USA, 1996. ACM.

[224] Haoqi Zhang, Edith Law, Rob Miller, Krzysztof Gajos, David Parkes, and Eric Horvitz. Human computation tasks with global constraints. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '12, pages 217–226, New York, NY, USA, 2012. ACM.

[225] Xing Zhang, Lijun Yin, Jeffrey F Cohn, Shaun Canavan, Michael Reale, Andy Horowitz, and Peng Liu. A high-resolution spontaneous 3d dynamic facial expression database. In *Automatic Face and Gesture Recognition (FG), 2013 10th IEEE International Conference and Workshops on*, pages 1–6. IEEE, 2013.

[226] C Lawrence Zitnick and Devi Parikh. Bringing semantics into focus using visual abstraction. In *Computer Vision and Pattern Recognition (CVPR), 2013 IEEE Conference on*, pages 3009–3016. IEEE, 2013.

[227] Matthias Zwicker, Mark Pauly, Oliver Knoll, and Markus Gross. Pointshop 3d: an interactive system for point-based surface editing. In *Proc. of the 29th annual conf. on Computer graphics and interactive techniques*, SIGGRAPH '02, pages 322–329, New York, NY, USA, 2002. ACM.