Exploring the World's Visual History

Ricardo Martin Brualla

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

Doctor of Philosophy

University of Washington

2016

Reading Committee:

Steven M. Seitz, Chair

Ira Kemelmacher-Shlizerman

David Gallup

Program Authorized to Offer Degree:
Computer Science and Engineering

University of Washington

Abstract

Exploring the World's Visual History

Ricardo Martin Brualla

Chair of the Supervisory Committee:
Professor Steven M. Seitz
Computer Science and Engineering

Collectively, every day we take hundreds of millions of photos of people, objects and places and share them in online services, like Facebook, Instagram, or Flickr. Together, these photos create a living visual record of the world that is growing every day and covers the whole planet. Now is the first time in history that we have access to more than a decade worth of such detailed visual information. In my thesis, I proposed novel techniques to analyze and visualize the world's history through the lens of these Internet photos.

First, I propose a method to synthesizing time-lapse videos of the world's most famous landmarks over several years from publicly available photos on the Internet. I processed a database containing 86 million photos and generated thousands of time-lapse videos that span up to a decade, and are effectively some of the longest time-series ever captured. The synthesized time-lapses show, for example, the retreat of glaciers, the construction floor by floor of skyscrapers, and seasonal changes in landscapes across the world. Furthermore, I extend the technique to create 3D time-lapses, where the virtual camera moves continuously in time and space, creating compelling parallax effects.

Next, I propose the *3D Wikipedia*, a system that analyzes online text together with online photos to automatically create interactive visualizations of famous landmarks that effectively convey their history. The system mines text and image co-occurrences across the Internet, to generate correspondences between objects described in the text and bounding boxes in the

3D model, that enable novel interactions for coordinated browsing of the reference text and the 3D model. Selecting discovered objects in the 3D visualization scrolls the text where the object is mentioned, and when clicking on discovered objects in the text, the camera moves to show the corresponding objects in the 3D model. In another mode, the text serves as a visual guide to the scene, where the visualization highlights the described objects as the user reads the text.

Finally, I propose a method to help visualize and analyze the millions of visits to tourist sites, by generating 3D reconstructions of large indoor spaces. These are a common failure case for Structure-from-Motion systems, that fail to generate a complete 3D model due to sparse coverage, and break them up instead into small, disconnected pieces. I jointly analyze Internet photos together with an annotated floorplan of the landmark to recover a 3D model of the landmark, where the disconnected 3D pieces are localized into the map's reference frame. My approach is akin to solving a 3D jigsaw puzzle, where the position and orientation of the 3D pieces is unknown. I extract position, orientation, and shape cues from the map and introduce a novel crowd flow cue between pieces that is based on how people travel between the rooms. The recovered complete 3D reconstructions allow mapping tourists' visits through the space, enabling compelling visualizations of their visits, and provide insights on tourists' spatio-temporal behaviors.

ACKNOWLEDGMENTS

First and foremost, I wish to thank my advisor Steve Seitz, who has taught me how to become a researcher, ask the right questions, methodically solve technical problems, and effectively communicate research ideas. This thesis would not have happened without his support and help over the years. I am also very thankful for all the great collaborators I had the pleasure to work with: Bryan Russell, David Gallup, Luke Zettlemoyer, Yanling He, and Ira Kemelmacher-Shlizerman. I also want to thank the "La Caixa" Foundation for their fellowship and economic support.

I would also like to thank my fellow students and postdoctoral researchers at GRAIL, whom I had hundreds of discussions about research and life over the years, including Aditya, Kathleen, Supasorn, Gilbert, Dan, Rahul, Qi, Neeraj, Richard, Alexander, and many others. Furthermore, I wish to thank the vibrant and welcoming UW CSE community, and my close graduate school friends Mark, Paris, Svet and Brandon, that helped make my graduate school experience much more enjoyable.

I also wish to thank the amazing mentors I have had over the years that helped paved my path towards my PhD: Albrecht Hess at the Deutsche Schule Madrid, Salvador Roura at the Universitat Politècnica de Catalunya, Juan Andrade Cetto at Institut de Robòtica i Informàtica Industrial, Enrique Alfonseca and Neil Alldrin at Google, and Cha Zhang at Microsoft Research. I can only hope that throughout my career, I will be able to give back such mentorship to the coming generations of scientists.

Finally, I wish to thank my parents and my family for their love and support.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

Chapter 1

INTRODUCTION

Within a lifetime, we and the world change dramatically. Children grow into adults. Cities expand into suburbs and old buildings are replaced with skyscrapers. Societies transform due to social movements and migratory fluxes. Our planet also changes, animals become extinct, forests shrink, and glaciers recede. Although we have lived through these changes, it is hard to convey many of these evolutionary changes to others. Thankfully, photography allows us to capture glimpses of ourselves and the world, that with time become objective memories of our past. Together, these personal photographs create a thorough record of our lives, documenting the visual history of ourselves, our family and friends, and the places we visit. Furthermore, in the Internet era, many of these personal photos are shared online, and collectively become an immense visual record of the world, that covers the last decade or so.

In this thesis, I explore the world's visual history through the lens of these Internet photos. The motivation for this is two-fold. First, the rate at which we are taking photos has increased dramatically in the past decade. Whereas at the beginning of the 1900s, people kept dozens of photographs taken on marked occasions in their lives by professional photographers, people today carry smartphones at all times that can take thousands of photos on a single battery charge at virtually no cost. Any famous place on Earth is photographed millions of times each year by tourists. Similarly, most people are photographed thousands of times per year. The evolution of children in particular is captured with minute details by their parents and family. Indeed, one study [51] estimated that more than one trillion personal photos would be taken in 2015. That corresponds to 150 photos per person in the planet in 2015, or a photo about every two days on average, with people in developed countries taking many more. This trend started with the mass adoption of digital cameras in the 2000s, and since

then, the number of photos taken each year has increased exponentially.

Parallel to this trend, the digital photography revolution has also allowed easier access to these photos. Whereas in the past personal photo collections were stored as physical prints and otherwise not easily accessible, now people share thousands of photos online. Internet photo-sharing started around 2005 with sites like Picasa, Panoramio, and Flickr, that allowed users to upload and share their personal photo collections online, and continued in social networks like Facebook, Instagram, and Snapchat, that dominate online photo-sharing nowadays. Over time these sites have accumulated billions of photos, that can be easily accessed over the Internet. For instance, the query "Colosseum in Rome" on Flickr returns more than 200,000 images of the monument from every possible angle and in any lighting condition.

Combined, these two trends mean that we now have access to a decade's worth of photos for most landmarks around the world. Can we visualize the history of a place by looking at these photos? How has the place changed over a decade?

1.1 Contributions

In this thesis, I develop new Computer Vision and Computer Graphics algorithms and techniques that enable new visualizations of the history of the world. My contributions are divided along three time scales: the history in the Internet era, the history of places and the history of visits to these places.

1.1.1 History in the Internet era

A common technique to visualize changes in a scene is rephotography [89]. Given an old photo of a scene, the rephotography technique consists of taking another photo of the scene from the exact same viewpoint. A viewer can easily see the changes in the scene when comparing the new photo to the old one. We can use a similar technique to understand changes in a famous landmark using Internet photos.

For example, when visiting the Briksdalsbreen Glacier in Norway (shown as of 2015 in

(a) Briksdalsbreen Glacier in 2005 (b) Briksdalsbreen Glacier in 2015

Figure 1.1: Two photos taken from a similar viewpoint of the Briksdalsbreen Glacier in 2005 and 2015. The side by side comparison shows the retreat of the glacier, whose lower half has disappeared. Photo credits: Wikipedia users Saperaud and Gonzo Lubitsch.

Figure 1.1(b)), one might wonder whether the glacier is receding and if so, how fast. To answer these questions, we can compare this photo with older photos taken from similar viewpoints, by querying for the glacier's name and a year number, like "Briksdalsbreen Glacier 2005". This query retrieves 81 photos in Flickr, with some of them approximately matching the viewpoint of the 2015 photo. The comparison shown in Figure 1.1 shows the dramatic retreat of the glacier in the past decade, whose lower half has disappeared. However, such one-off comparisons are not sufficient to fully understand the glacier's retreat. Did the glacier disappear all at once or slowly over the last ten years? Did the glacier grow at some moment in the past decade? These questions can only be answered when analyzing the extent of the glacier in many photos through time. This is very challenging due to the variability of Internet photos, that are taken from different viewpoints, with different cameras, in different lighting conditions and contain objects posing in the foreground, which create distractions in any visualization of the collection over time. Furthermore, the photos' timestamps or metadata are not reliable, and can be off by several years.

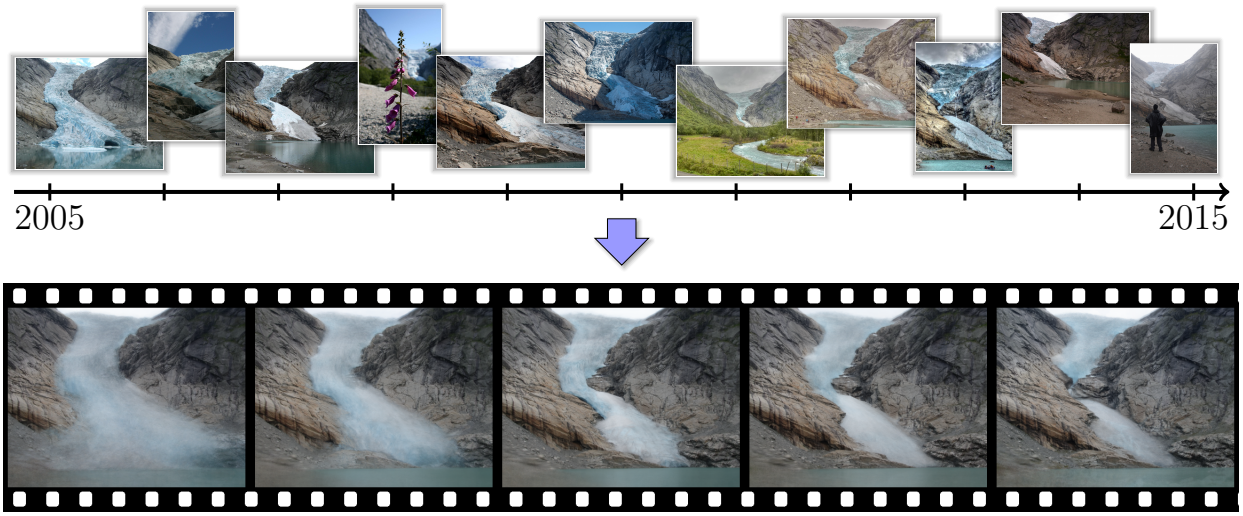


Figure 1.2: I mine Internet photo collections to generate time-lapse videos of locations all over the world. The time-lapses visualize a multitude of changes, like the retreat of the Briksdalsbreen Glacier in Norway shown above. The continuous time-lapse (bottom) is computed from hundreds of Internet photos (samples on top). Photo credits: Flickr users Aliento Más Allá, jirihnidek, mcxurxo, elka_cz, Juan Jesús Orío, Klaus Wißkirchen, Daikrieg, Free the image, dration and Nadav Tobias.

In Chapter 2, I propose a method to reveal the visual history of the world, that uses Internet photos to generate time-lapse videos of famous landmarks, as shown in Figure 1.2. The technique starts by recovering a 3D reconstruction and finding popular viewpoints of the site. Then, it sorts the photos by date and warps each photo onto a common viewpoint. Finally, the appearance of the sequence is stabilized to compensate for lighting effects and minimize flicker.

I processed a dataset of 86 million publicly available photos from Picasa and Panoramio and produced more than 10,000 time-lapses around the world. The resulting time-lapses show diverse changes in the world's most popular sites, like glaciers shrinking, skyscrapers being constructed, and waterfalls changing course. Furthermore, in Chapter 3, I describe

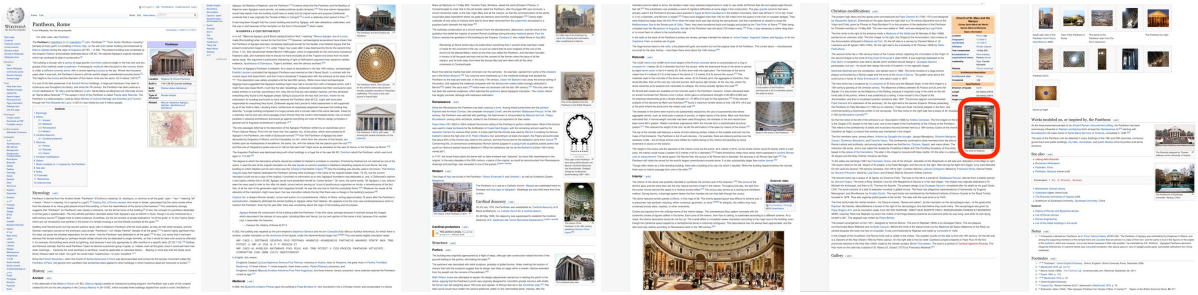


Figure 1.3: Six pages of Wikipedia article for the Pantheon in Rome. Despite containing more than 30 descriptions of objects present in the scene, only one, the tomb of Rafael, is depicted among the 22 photos shown in the article (highlighted in red).

an extension of the technique to synthesize 3D time-lapse videos, where the camera moves continuously in space and time. The extended technique is able to recreate cinematographic effects often used by professional film makers, like camera orbits or push movements, that add depth and dramatism to the time-lapse videos.

1.1.2 History of a place

Internet photos uploaded by tourists only go about a decade or so in time. How can we learn about the history of a place like the Pantheon in Rome, that is 2000 years old? Only a part of the history of the Pantheon is captured in photos; most of its history is instead carefully documented in history books. For instance, the Wikipedia article of the Pantheon (Figure 1.3) is filled with historical facts and descriptions of all artifacts in the building. The text guides the reader in a clockwise manner around the interior, describing every significant painting or statue and their history. Nonetheless, it is hard for a reader to develop an intuition of the space, as there are few images depicting the mentioned objects. Indeed, in the Wikipedia article for the Pantheon, there is only one photo ("Raphael's tomb") for the 30 described objects in the text.

On the other hand, the Pantheon has been photographed millions of times by tourists

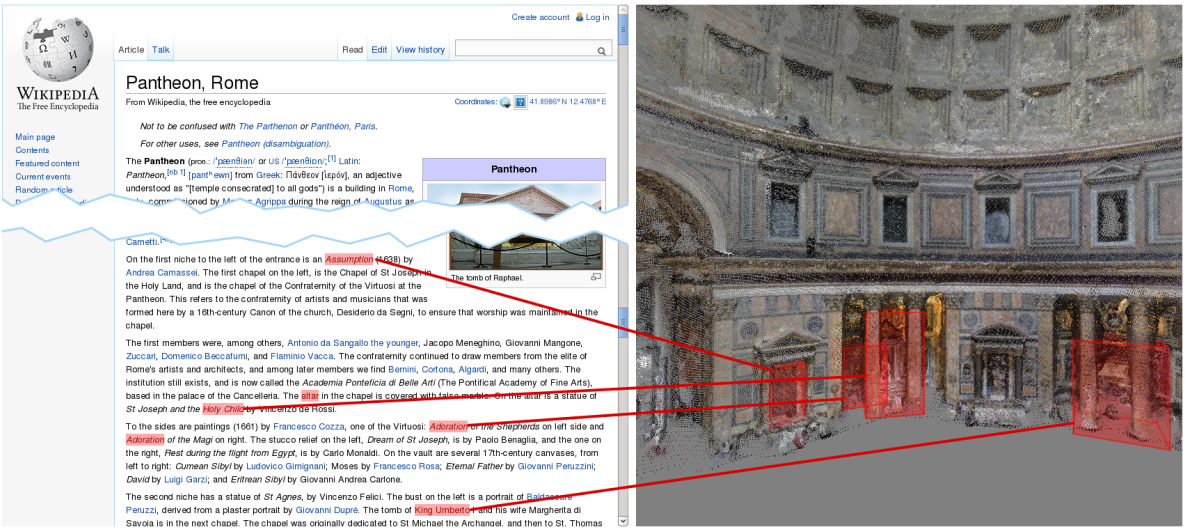


Figure 1.4: Given a reference text describing a specific site, for example the Wikipedia article above for the Pantheon, we automatically create a labeled 3D reconstruction, with objects in the model linked to where they are mentioned in the text. The user interface enables coordinated browsing of the text with the visualization.

around the world, including thousands of perfectly cropped photos of each of the individual objects contained within, together with corresponding captions describing the objects. Can we combine the historical facts in the text with the dense image coverage provided by Internet photos? If so, can we create an improved, more visual version of Wikipedia, that leverages these dense photos?

Chapter 4 describes the *3D Wikipedia*, an interactive visualization of landmarks and their history. The visualization, depicted in Figure 1.4, displays on one side an authoritative text of the site, like a Wikipedia article, and on the other, a photorealistic 3D model of the site. The system extracts correspondences between the text and the 3D model by mining text and image co-occurrences across the whole Internet using Google Image Search. The discovered correspondences enable novel interactions that allow joint navigation of both the text and the 3D model. For example, when selecting an object in the text, the 3D visualization smoothly

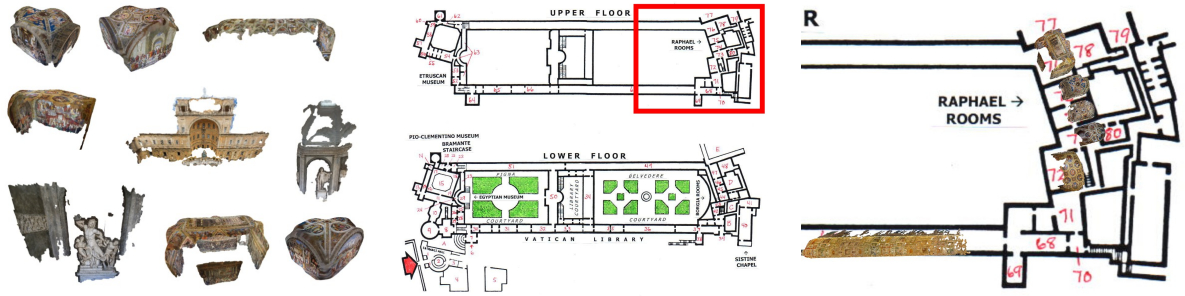
Figure 1.5: Photos of a visit to the Vatican Museums. The visit follows a typical route through the Museums: starts in the entrance, then goes into Cortille della Pigna, and Cortile Ottagono, followed by the Hall of Maps and into the Salla dell'Immacolata, Raphael Rooms and finally the Sistine Chapel, shown from left to right.

moves the camera to show its location in the site. In a similar manner, when clicking on an object in the 3D visualization, the text scrolls to where the object is mentioned. In another mode, the visualization follows a user reading through the text, showing the described objects as they are mentioned in the text.

1.1.3 History of a visit

Another interesting aspect of tourists sites is the way people visit them and the spatial trajectory of their visits. The paths of thousands of tourists across a site can reveal important information about a place, like the popular spots within the landmark, the areas that can be accessed or the time it takes to visit a site. This behavioural information can be very useful to recreate the experience of visiting a place, or to help a tourist plan the best path through a landmark. To recover the paths tourists take through the Vatican Museums (Figure 1.5), we analyze the photos they shared online of their visit. Each photo serves as a clue, that together with its timestamp, reveals when and where the tourist was throughout their visit.

Nevertheless, it can be hard to know where each photo was taken within the Vatican Museums. GPS does not work indoors, and traditional Structure-from-Motion techniques [5, 105, 117] fail to reconstruct a single complete model of the site from Internet photos, due to



(a) 3D pieces (b) Annotated map (c) Reconstructed pieces

Figure 1.6: The *3D Jigsaw Puzzle*: Given (a) a set of disconnected reconstructed 3D models of a large indoor scene, for example the Vatican, I jointly reason about (b) a map of the site and the 3D pieces to produce a globally consistent reconstruction of the entire space. (c) shows a close-up of pieces automatically registered with the top-right part of the map.

sparse coverage and missing matches across rooms, and instead generate many disconnected 3D pieces. In Chapter 5, I tackle this problem and introduce the *3D Jigsaw Puzzle*, a system to reconstruct large indoor spaces. The proposed method takes Internet photos of the site, together with an annotated map, such as the ones in guidebooks, and automatically lays out the 3D pieces in a global coordinate system provided by the map, as shown in Figure 1.6. It leverages position, orientation, and shape cues extracted from the map to recover the global layout of the pieces. Most interestingly, I propose a crowd flow cue that measures how people move across the pieces and disambiguates their relative orientations.

The rest of this thesis is structured as follows. Chapter 2 proposes a method to mine time-lapse videos from Internet photos. The method is extended in Chapter 3 to generate 3D time-lapses, where the virtual camera moves continuously both in space and time. Chapter 4 presents the *3D Wikipedia* to visualize the history of a place described in online sites like Wikipedia. The *3D Jigsaw Puzzle* is described in Chapter 5 and is a key step into understanding people's visits to large indoor tourist sites. Chapter 6 concludes the thesis and discusses future work. In the rest of the introduction, I review previous approaches to

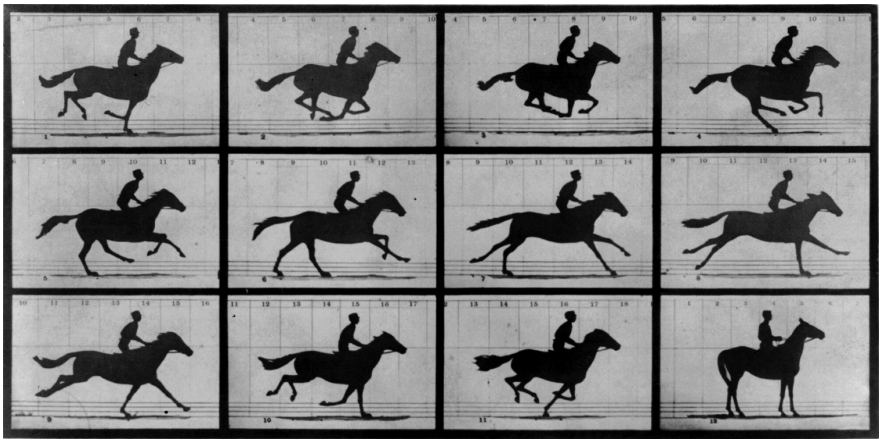


Figure 1.7: Sequence of photos of galloping horse by Eadweard Muybridge in 1878 that solved the mystery of whether a horse ever lifts all its hoofs off the ground while galloping (see middle two photos in top row).

time visualization.

1.2 Related Work

The topic of time visualization has a rich history. For thousands of years, historical events and biblical stories have been portrayed by artists in paintings and sculptures, that show multiple scenes of the same story next to each other, like the Trajan's Column in Rome (ca. 106-113 CE), whose long frieze depicts 155 scenes from the Dacian wars [24], or the Sistine Chapel's ceiling (1508-1512 CE), that contains nine scenes of the Book of Genesis [85]. However, ever since the invention of the daguerreotype in the 1830s, artists and scientists have used photography to visualize events at time-scales different from what humans perceive.

On one hand, scientists have often desired to freeze time, in order to visualize and study phenomena that happen too fast for our eyes to see, like the motion of a galloping horse or glass shattering. Although today we are used to snapshots, that capture scenes frozen in time, this was not the case at the beginning of photography, when daguerreotype processes required long exposures of up to 10 minutes, while the subjects had to stay still. The breakthrough

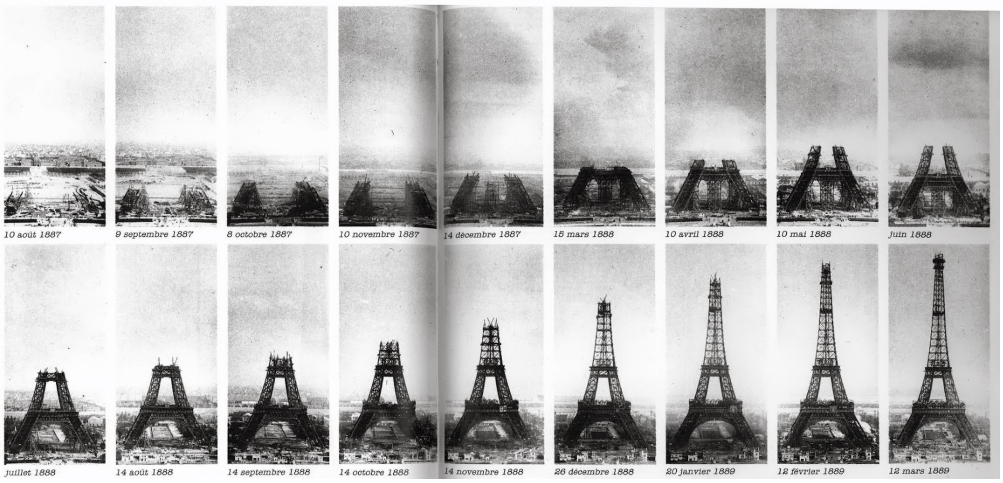


Figure 1.8: Time-lapse sequence of construction of the Eiffel Tower in Paris, France between 1887 and 1889.

works of Muybridge and Marey [73, 79] in the late 1880s enabled the capture of motion, to study human and animal locomotion (see Figure 1.7). Over time, new techniques allowed imaging much faster events, in the order of microseconds, like stroboscopy and ultra-fast flash photography [29]. Photographing smaller time-scales is still an active area of research, with recent systems using laser pulses in the order of femtoseconds, e.g. 10^{-15} seconds, that capture precisely how light travels through a scene [112].

Another interesting topic is visualizing phenomena occurring too slow for humans to perceive, from things happening in minutes, like the clouds rolling or the sun setting, to centuries, like sequoias growing or cities expanding. Time-lapse photography [82, 109] consists of placing a static camera that captures frames at specific time intervals, to create sped up movies of slow processes, such as the blooming of flowers or the construction of a building. Figure 1.8 shows an early example of time-lapse photography, depicting the construction of the Eiffel Tower over two years. However, placing a camera for long enough to visualize changes in the scene is often logistically complicated, like when visualizing the retreat of glaciers, that happen over periods of several years in very remote areas [28]. Another ap-

proach is rephotography [45, 89], that involves taking a repeat photograph from the exact viewpoint of an older one, to visualize the changes in the scene over large periods of time. Nevertheless, visualizing changes using rephotography can be challenging, due to vastly different appearances between the photos. This happens often when the photos are taken with different photography process, like digital versus chemical film processes, in different lighting conditions, or at different times of day.

In this thesis, I am inspired by rephotography and time-lapse techniques to visualize changes over time. Instead of capturing repeat photographs, I seek to find places where enough photos have been captured over time, that provide a dense temporal sampling of the scene over long periods of time. These places correspond to tourists hot spots, that attract millions of tourists every year that take photos of the landmark and share them online. By mining millions of these Internet photos, I set to visualize changes in these famous landmarks all around the world.

However, photos shared online lack precise localization information, and automatically understanding what is photographed in each photo is challenging. Throughout my thesis, I build upon key breakthroughs in registering these unstructured photo collections [5, 105], that recover for each photo, the location where it was taken, together with a 3D model of the scene. Indeed, state-of-the-art Structure-from-Motion systems generate 3D reconstructions from millions of photographs in several hours on a single machine [33, 49]. Previous work [77, 94, 95] has addressed generating spatio-temporal reconstructions from unstructured photo collections, like Internet and historical imagery. However, these works generate 4D reconstructions of the world that are sparse, composed by 3D points and 3D planar patches, that fall short of depicting the complete scene over time. In my thesis, I build upon Multi-view Stereo techniques [35, 96] and develop new stereo methods, to generate photorealistic visualizations of the history of a place. Section 2.1 provides a more in-depth review of these works and other related computational techniques to time-lapse photography.

Chapter 2

TIME-LAPSE MINING FROM INTERNET PHOTOS

We see the world at a fixed temporal scale, in which life advances one second at a time. Instead, suppose that you could observe an entire year in a few seconds—a 10 million times speed-up. At this scale, you could see cities expand, glaciers shrink, seasons change, and children grow continuously. Time-lapse photography provides a fascinating glimpse into these timescales. And while limited time-lapse capabilities are available on consumer cameras [7, 52], observing these ultra-slow effects requires a camera that is locked down and focused on a single target over a period of months or years [28].

Yet, these ultra-slow changes are documented by the billions of photos that people take over time. Indeed, an Internet image search for any popular site yields several years worth of photos. In this chapter, we describe how to transform these photo collections into high quality, stabilized time-lapse videos. Figure 2.1 shows a few frames from one result video of a glacier receding over a decade. This capability is transformative; whereas before it took months or years to create one such time-lapse, we can now almost instantly create thousands of time-lapses covering the most popular places on earth. The challenge now is to find the interesting ones, from all of the public photos in the world. We call this problem *time-lapse mining*.

Creating high quality time-lapses from Internet photo sharing sites is challenging, due to the vast viewpoint and appearance variation in such collections. The main technical contribution of this chapter is an approach for producing extremely stable videos, in which viewpoint and transient appearance changes are almost imperceptible, allowing the viewer to focus on the more salient, longer time scale scene changes. We employ Structure-from-Motion

---

This chapter describes work that was originally published in ACM SIGGRAPH 2015 [75].

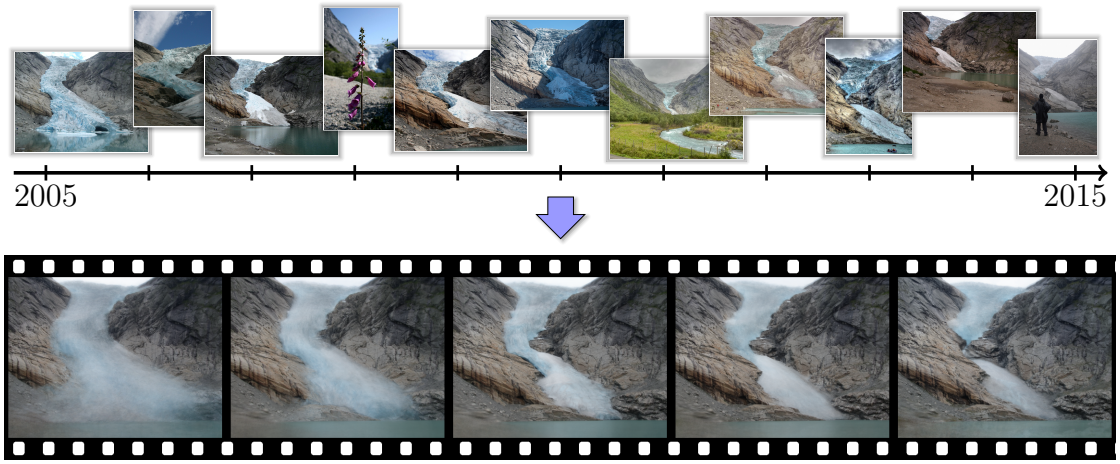


Figure 2.1: We mine Internet photo collections to generate time-lapse videos of locations all over the world. Our time-lapses visualize a multitude of changes, like the retreat of the Briksdalsbreen Glacier in Norway shown above. The continuous time-lapse (bottom) is computed from hundreds of Internet photos (samples on top). Photo credits: Flickr users Aliento Más Allá, jirihnidek, mcxurxo, elka_cz, Juan Jesús Orío, Klaus Wißkirchen, Daikrieg, Free the image, dration and Nadav Tobias.

and stereo algorithms to compensate for viewpoint variations, and a simple but effective new temporal filtering approach to stabilize appearance. Our second significant contribution is a world-scale deployment, where we process over 80 million public Internet photos, yielding several thousand mined time-lapses spanning the worlds most photographed sites.

2.1 Related work

This section is divided in three parts. First, we review previous works that generate temporal models and visualizations from unstructured photo-collections, that we denote *unstructured time-lapses*. Then, we survey relevant work to traditional time-lapse photography within the areas of Computer Vision and Graphics. Finally, we discuss some applications of time-lapse photography in science.

(a) Reference image (April 2011) (b) Recovered appearances

Figure 2.2: Recovered appearances at different points in time of a wall in 5 Pointz in Queens, New York, by Scene Chronology [77]. Left: a sample image and the location of a space-time cuboid. Right: different appearances over time of the space-time cuboid.

2.1.1 Unstructured time-lapses

In Scene Chronology, Matzen and Snavely [77] discover changing elements in 3D scenes by clustering reconstructed 3D patches into space-time cuboids. The authors estimate the period of time an element was visible in the scene and propose a 3D visualization where the user can move through time and space, that only shows the discovered elements that existed at the given time. Their approach is limited however to reconstructing planar structures, like billboards or graffiti in urban scenes. Figure 2.2 displays different recovered appearances of the same space-time cuboid in a graffiti exhibit.

The 4D Cities project [95, 94] models changes in a city using historical imagery over several decades. The authors propose a probabilistic model of the time span of reconstructed 3D points, and find approximate solutions using Markov Chain Monte Carlo (MCMC). By reasoning about the visible points in a historic photograph, they can infer its timestamp, oftentimes improving the accuracy over its existing timestamp, if available. The generated 4D representation is a time-varying point cloud, whose 3D points appear and disappear over time, as shown in Figure 2.3

(a) Reference image (b) 1960

(c) 1965 (d) 1970

Figure 2.3: 4D model of the city of Atlanta generated from the 4D Cities project [95, 94]. a) shows a reference image of the downtown Atlanta and b), c) and d) 3D point cloud at different time periods.

Our approach substantially differs from both projects in key aspects. Our focus is to generate complete time-lapse videos (with no holes), instead of building sparse 4D representations of the world. Our approach also works on a global scale, discovering thousands of time-lapses all over the world, in contrast to the handful of sites analyzed by previous work. Finally, our system is not limited in scope to urban scenes and we generate time-lapse videos for diverse natural phenomena.

Lastly, Picasa FaceMovies [59] use a personal photo collection to generate a movie of how a person ages through time. The approach detects a person's face in the whole photo collection, and then computes a subsequence of the photos that limits changes in pose and expression. The resulting FaceMovie shows one photo at a time, with the subject's face centered in the frame, but fails to create the illusion that time flows continuously.

2.1.2 Standard Time-lapses

A common approach to visualizing how a scene changes over time is rephotography, where one compares two photos taken from the same viewpoint. Bae *et al.* [9] introduced an user interface to guide a photographer to lock in the viewpoint of an old historical photo of the scene. This is challenging due to drastic changes in appearance, and automatic methods often fail to register historical photos with the current ones. The help the user, the system first builds a two-view 3D reconstruction of the scene using, and then the user marks correspondences between the old photo and the current views. This enables the system to guide real-time the user to re-capture the historical photo from the same viewpoint. A similar approach was taken by *ConstructAide* [57], a system that aids in analyzing and visualizing construction progress by having the user guide the registration of unstructured photos to a 3D model of a building.

The synthesis of time-lapse videos from static video cameras, like webcams, has been explored in the literature. In Computational Time-lapse Video [12], ordinary videos are condensed into time-lapses, using sampling and filtering strategies to convey different visual objectives. Rubinstein *et al.* [90] propose a method to denoise small motions in a time-lapse, optimizing the resulting video by borrowing pixel values in a spatio-temporal neighbourhood. Motion denoising is formulated as a 3D Markov Random Field that optimizes temporal smoothness of the output video by computing spatio-temporal displacements of output pixels with respect to the input. Their system is constrained by the optimization and does not scale beyond short, low resolution sequences.

Static time-lapse videos provide extensive information about the scene appearance under different lighting conditions. This has been exploited to compute factored lighting models [108] and perform photometric stereo to obtain scene BRDFs [2]. Scene geometry can also be inferred from the shadows cast by clouds [1, 53]. By using a database of time-lapse videos, [66, 98] learn appearance transfer models that can change the time of day or time of year of a photograph.

Finally, high quality capturing static camera time-lapses is also challenging. Kinsman [60] provides a thorough guide to time-lapse photography. The author provides insights on how to best prepare certain types of scenes, from growing plants that require heating and very specific lighting, to construction sites, where the camera needs to be properly secured and weather-proofed. Furthermore, auto-exposure settings cause flickering artifacts in many sequences, that need to be removed with specialized software [114]. The author points other hardware limitations, like the reliability of shutters in DLSR cameras, that do not last more than a few hundred thousand shots. Another challenge is the poor accuracy of the cameras' internal clocks, that Welty *et al.* [115] show can drift up to five minutes in one year, and limit their use in long term scientific studies.

2.1.3 Time-lapse Photography in Science

Time-lapse photography has been used in many fields of science to measure visual quantities over time. For example, the Extreme Ice Survey [28] is a long-term time-lapse photography project that monitors the retreat of glaciers around the world. Their approach consists of placing cameras in remote spots overlooking a glacier, where the camera is completely autonomous, in a weather-proof case, and connected to solar panels to replenish its batteries. In [6] scientists compute the flow speed of glaciers using the Extreme Ice Survey data, that ranges from 3 to 50 meters per day and varies over time. For instance, glaciologists are specially interested in glacier *surges*, that are short events, where a glacier might advance at speeds up to 100 times faster than normal. Although approaches to glacier monitoring using Internet photos have limited spatial and temporal resolution compared to traditional scientific settings, they might enable a posteriori studies of events in places that were previously not monitored.

In a different domain, Lang and Hogg [68] measured the orbit of the Comet Holmes as it passed around the Sun in 2010 using astronomical pictures found on the Internet. The authors obtained photos of the comet querying Yahoo image search, calibrated them astronomically using neighbouring star locations, and recovered the location of the comet

in each photo. Finally, they infered the trajectory of the comet using a probabilistic model that reasoned about the location of the comet and the photos' timestamps. The recovered orbit aligns well with the best known trajectory of the comet, proving the validity of their method.

2.2 System overview

The input to our system is a collection of 86 million timestamped and geotagged photos around the world. The system automatically discovers all locations in the world with enough imagery and generates a time-lapse video for each.

Section 2.3 describes how candidate time-lapse video locations are mined from unstructured photo collections. Each candidate time-lapse video consists of a reference camera viewpoint and a set of nearby images. Next, the images of each candidate time-lapse are ordered chronologically and warped into the reference camera to compensate for viewpoint differences, as explained in Section 2.4. Section 2.5 describes our approach to stabilize the appearance of the video to compensate for varying lighting conditions and occlusions from transient objects like people.

2.3 Locating time-lapses at planet scale

In this section we present a method to discover locations for mined time-lapse videos. These locations correspond to camera viewpoints that, due to the prominence of the depicted scenes, have been photograped from a similar viewpoint repeatedly over time by many different tourists.

We pose the problem of discovering time-lapse viewpoints as finding clusters of images that feature the same subject from similar viewpoints. We first cluster the photos based on their geolocations into *landmarks* and for each landmark we compute 3D reconstructions using Structure-from-Motion techniques [3]. Note that a landmark may have several disjoint reconstructions, e.g., inside vs. outside.

To find popular viewpoints within a 3D reconstruction, we use the canonical view approach of [102]. Their approach works by analyzing SIFT feature co-occurrences to partition the set of images into groups of photos with similar content and viewpoint. Representative images are then chosen for each group, by finding images that share the most features within the group. We compute the 20 highest ranked reference images (canonical views) for each 3D model.

For each reference image I_R, we find "nearby" images $\{I_i\}$ with similar viewpoints and directions, satisfying the following criteria:

- the optical axis is within α degrees of the reference viewpoint direction and,
- the camera center is located within a radius $R = tan(\alpha) \cdot \overline{d}$ of the reference image camera center, where $\overline{d}$ is the average distance from the reference camera center to 3D locations of image features visible in the reference image,

where α is a camera inclusion threshold.

Finally, we filter all candidate time-lapses that contain fewer than 300 timestamped images. Note, that two different candidate time-lapses from the same landmark might overlap in the photos they include.

Figure 2.4 shows the discovered reference image of the Briskdalsbreen Glacier time-lapse and the camera centers of the nearby images as green points. Note the tongue of the glacier being occluded by the landscape in the bottom right image, which is discarded by our proximity constraint.

2.4 Geometric stabilization

In this section we describe how to correct the photos for different viewpoints with respect to the reference image. If the scene is nearly planar, a homography could be sufficient to warp image I_i into reference image I_R. We can compute such homographies by using RANSAC on projections of the 3D tracks in the SfM model from camera C_i to camera C_R. This baseline

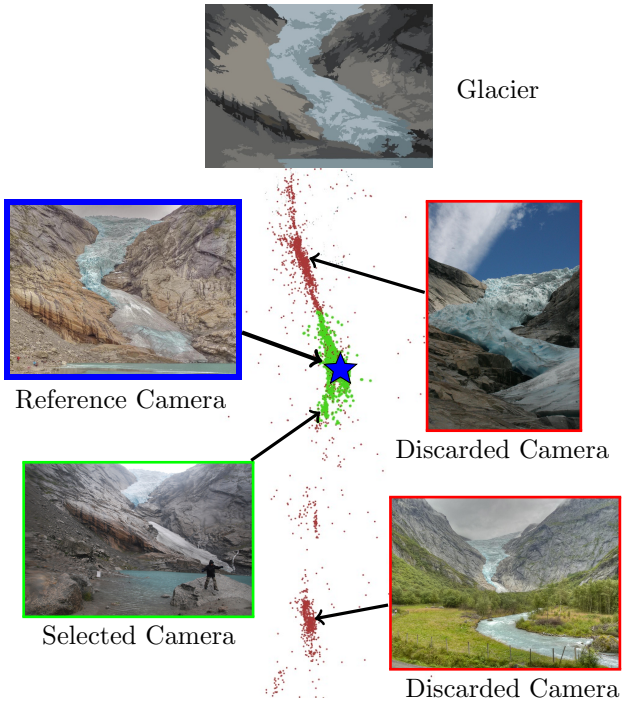


Figure 2.4: Top-down view of the Briksdalsbreen Glacier reconstruction. Red and green points correspond to the 9411 camera centers in the SfM reconstruction. The reference image for the time-lapse in Figure 2.1 is shown in top left and the blue star represents its camera center. Selected cameras for the time-lapse are shown in green and discarded cameras in red. The two images on the right correspond to other clusters in the distribution of photos of the scene. Photo credits: Flickr users Daikrieg, jirihnidek and Nadav Tobias.

method works well for scenes without parallax, but as expected, is not able to stabilize scenes with larger depth variations.

To account for parallax, we compute a depthmap D from the viewpoint of the reference image I_R. Although changes in the scene geometry over time are not modeled, we found that computing one global depthmap for the whole sequence provides adequate alignment for most scenes.

We use a temporal version of the classical plane sweep multi-view stereo algorithm [56],

modified to account for changing scene geometry and occluders like people. The main idea is to compute matching costs between images that were taken close in time. As with classical plane sweep, we generate a set of fronto-parallel depth planes with respect to the reference image I_R to compute matching costs. We discard the nearest and farthest 1% of 3D SfM points as well as 3D points with triangulation angles of less than 2 degrees, and evenly distribute enough depth planes (with a maximum of 200) over the depth range of the remaining 3D points to cover all disparity values.

We now define our temporal matching cost. Traditionally, stereo methods choose a reference image and only compute matching costs against that image. This does not work for our scenes as the scene geometry is changing over time. Instead, we compute a matching cost for each image as reference, using only images with nearby timestamps for matching, and then compute the overall cost as the median of costs over time, as described next.

Given the sequence of input images $(I_1, \ldots, I_n)$ ordered by timestamp, the per-image cost $C_d^i(p)$ for pixel p at depth d at timestamp i is defined as

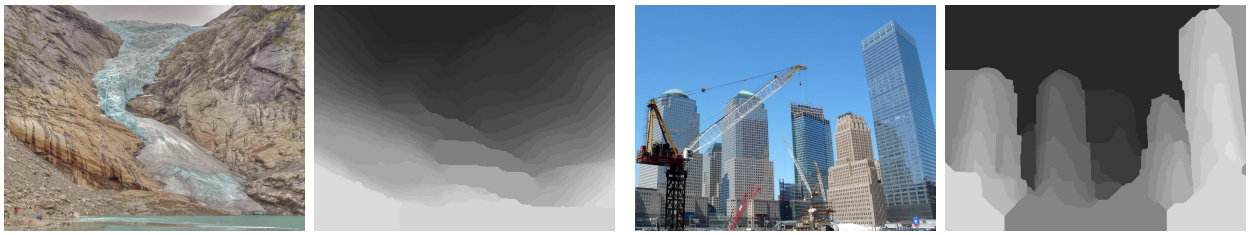
$$C_d^i(p) = \mathbf{median}_{j \in [i-T, i+T]} NCC_d(i, j, p) \tag{2.1}$$

where $NCC_d(i, j, p)$ is the normalized cross correlation of a patch of size $K = 7$ around p of the projections of images i and j to the depth plane d, and $T = 20$ is the temporal window size. The overall cost is then

$$C_d(p) = \mathbf{median}_{i \in [1,n]} C_d^i(p). \tag{2.2}$$

We compute a smooth depthmap D by using a standard MRF formulation where the data term for each plane is the matching cost C_d described above and the spatial term is a truncated L_1 distance [16]. We used a spatial term weight of 0.2 and a truncation parameter of 4 disparity values. Figure 2.5 shows the resulting depthmaps for two sites.

Finally, we compute the warped images I_i^w by projecting each image into the reference camera C_R. For each pixel in I_R, we find its correspondence in I_i by using the depthmap to infer its 3D position and projection into I_i, using z-buffering to account for occlusions. We inpaint occluded pixels whose projection falls inside the image boundary of I_i using [110].



(a) Briksdalsbreen Glacier, Norway (b) Goldman Sachs Tower, New York

Figure 2.5: Reference image and computed depthmap for Briskdalsbreen Glacier and Goldman Sachs Tower scenes. Lighter colors represent pixels closer to the camera. Note that in the Goldman Sachs Tower scene, the building under construction is reconstructed even though it is absent for part of the time-lapse. Photo credits: Flickr users Daikrieg and Cebete.

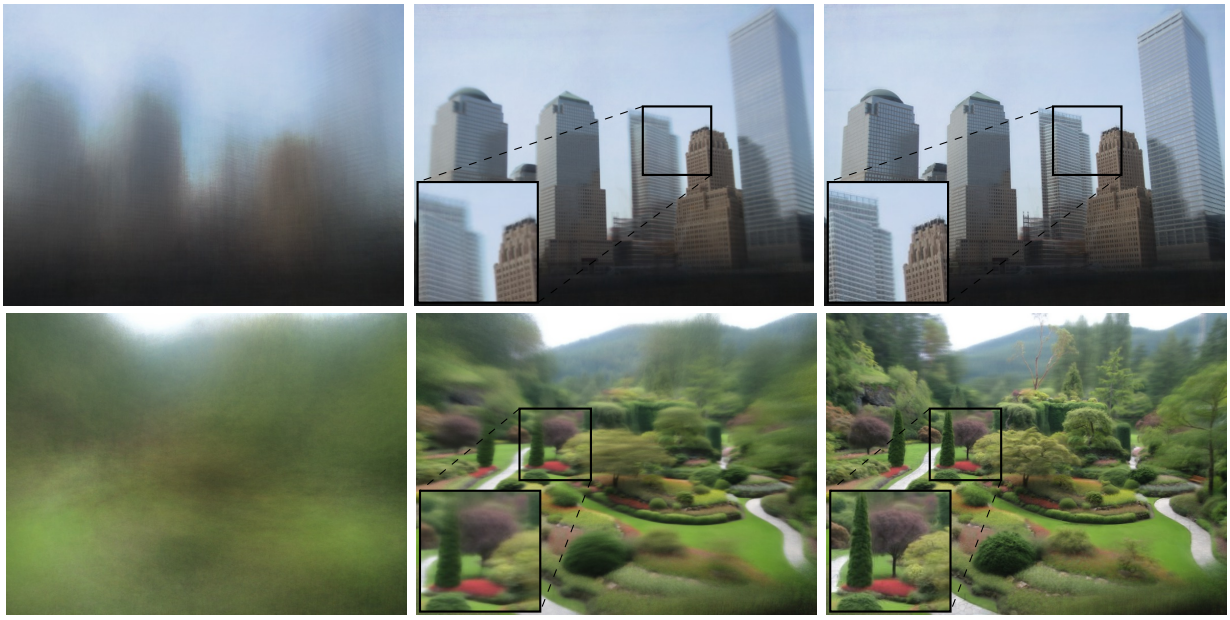
Figure 2.6 compares stabilization techniques. We test two methods, stabilization with homographies and the proposed stabilization with stereo, and compute for each the median image of the stabilized sequence. We also show the median of all input images (without stabilization) for comparison. The stereo method produces significantly sharper results.

2.5 Appearance stabilization

In this section we describe how to stabilize the appearance of the warped images to correct for different lighting conditions and occluders. We formulate this task as computing an output time-lapse video frame for each warped image $I_i^w(p)$.

One effective approach for removing noise is median filtering. Bennett and McMillan [12] apply a temporally moving median filter to the frames of a time-lapse video. We adapt this method to the warped image sequences by computing the median of the valid pixels in the warped images, i.e., the pixels whose projection into the input image camera lies within the image frame. We found that large temporal windows are needed to reduce flicker but also result in oversmoothed transitions.

To address this drawback, we introduce a new temporal regularization approach. For



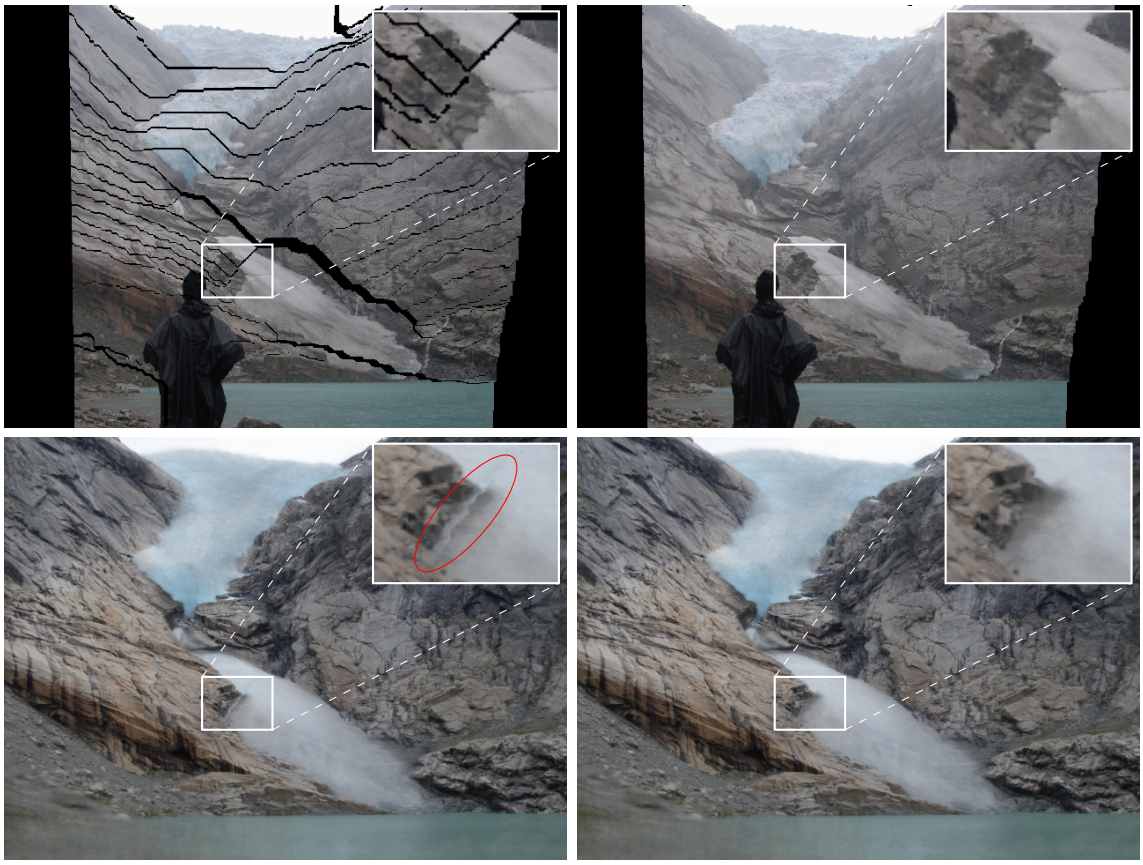
(a) No alignment (b) Homography alignment (c) Stereo alignment

Figure 2.6: Stabilization results for two different scenes, Goldman Sachs Tower (top) and Butchart Gardens (bottom). Aligning the images with depth (c), produces a sharper composite compared to homography (b), or no alignment (a).

each pixel, the goal is to compute its RGB value over time, by regularizing the pixel values of the warped sequence. Let $x_i = I_i^w(p) \in [0,1]^3$ be the RGB value in the warped image i, and let y_i be the RGB value in the output frame that we wish to compute. We optimize the following:

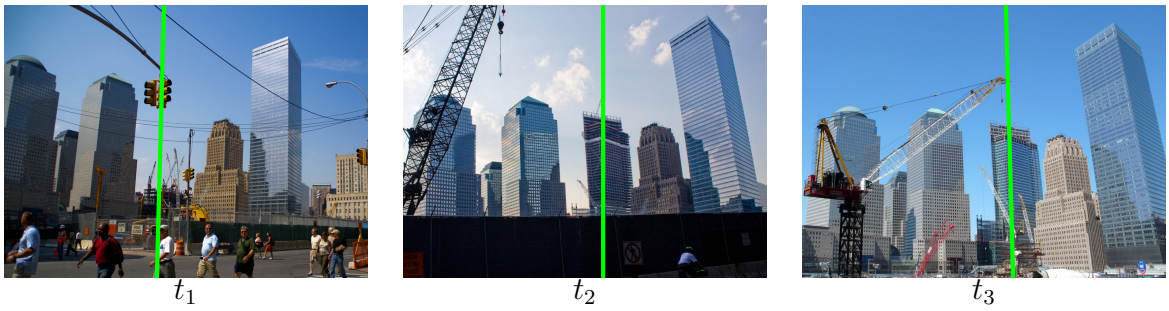
$$\min_{y_1,\ldots,y_n} \sum_{i|x_i \neq \emptyset} \delta\left(\|y_i - x_i\|\right) + \lambda \sum_i \delta\left(\|y_{i+1} - y_i\|\right) \tag{2.3}$$

where $\delta(\cdot)$ is a loss function, λ is a temporal smoothing coefficient and $x_i = \emptyset$ when p corresponds to pixel coordinates outside the image boundary of I_i, i.e., has no correspondence in I_i. As for occluded pixels, we found that inpainting them works better than treating them as missing because they appear consistently around depth discontinuities and our temporal regularization operates only on a per-pixel basis, i.e., lacks a spatial regularization term. Figure 2.7 shows the effects of inpainting in the warped images and the resulting artifacts in

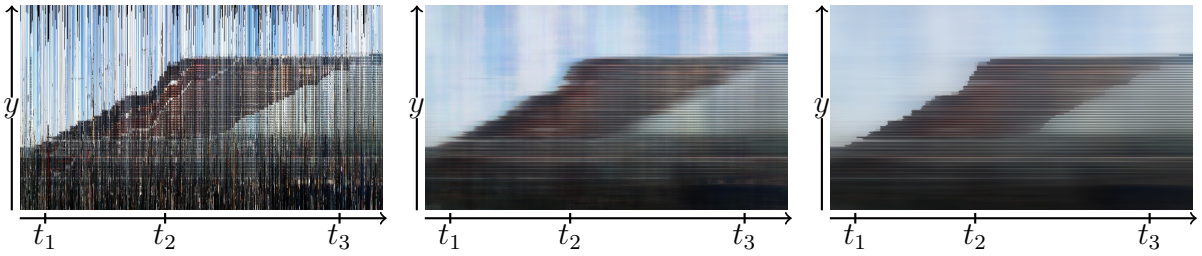


(a) Without inpainting (b) With inpainting

Figure 2.7: Effects of inpainting in the resulting time-lapses. Top: Warped images without inpainting (left) and with inpainting (right). Occluded pixels and pixels outside the input frustrum are shown in black. Note only occluded pixels are inpainted. Bottom: Corresponding frames of the resulting time-lapses. Note the artifacts around the depth discontinuities without inpainting. Photo credits: Flickr user Nadav Tobias.



(a) Images at different stages of construction



(b) Warped image sequence (c) Moving median (d) Regularized

Figure 2.8: Appearance stabilization for the Goldman Sachs Tower scene. (a): 3 sample images of the sequence showing the building at different stages of construction, with the pixel column of the y-t profiles highlighted. (b): y-t profile of the warped image sequence, showing this pixel column over time (moving to right). (c): result of temporal moving median filter of width 80 [12]. (d): result of our proposed temporal regularization with smoothing coefficient $\lambda = 100$. Moving median blurs the transitions and has more flickering, particularly in the sky pixels. Photo credits: Flickr users Zack Lee, ToastyKen and Cebete.

Figure 2.9: Map of the location of discovered time-lapses. Europe contains the highest density of time-lapses, while few exist in Africa and South America, as there are fewer photos available.

the output frames around depth discontinuities if not used.

We experimented with several loss functions, including L_1 and L_2. We found L_2 works best for smooth transitions, whereas L_1 behaves better at discontinuities; we obtained best results with Huber, a robust loss function that is L_2 near 0, and L_1 elsewhere. Figure 2.8 compares a moving median with our Huber approach and shows the advantage of our method (easier seen in the video).

2.6 Planet scale time-lapse results

We mined time-lapses from 86M public geolocated photos from Picasa and Panoramio. We clustered 120K different landmarks and computed 755K 3D reconstructions. We then discovered 10,728 time-lapses across 2942 landmarks, that contain more than 300 images, using the camera selection criteria of $\alpha = 10$ degrees. We mined the time-lapses on a cluster with over 1000 nodes.

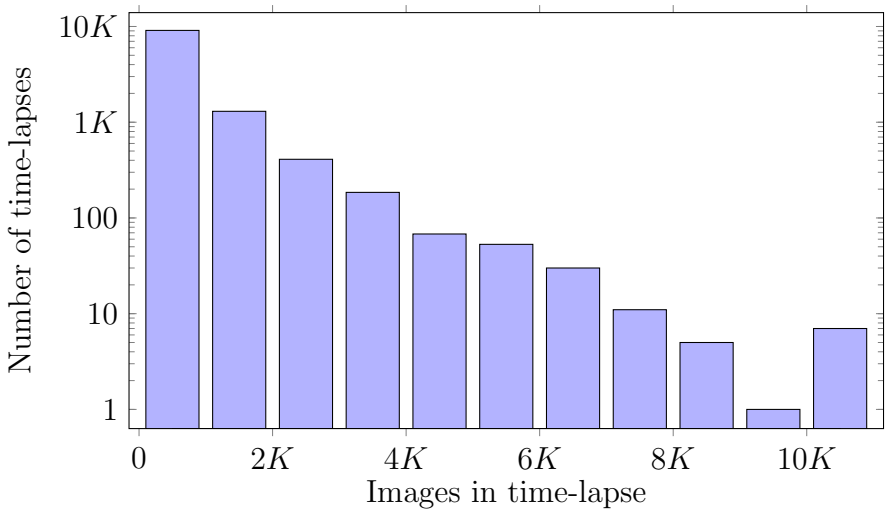
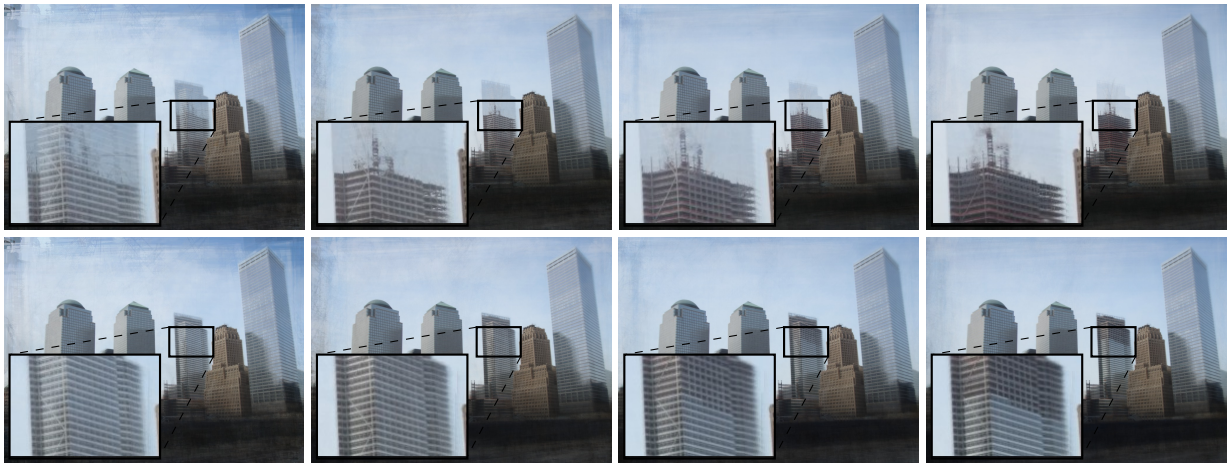


Figure 2.10: Histogram of number of cameras in the discovered time-lapses, ranging from 300 to 10953 photos.

Figure 2.9 shows that the discovered time-lapses cover the globe and follow a similar distribution as publicly available Internet photos [48]. Figure 2.10 shows a histogram of the length of the discovered time-lapses. A view of London from Greenwich Park contains several of the longest time-lapse sequences, with more than 10K photos each. The quality of the resulting time-lapse videos improves considerably with increasing number of input photos, as shown in Figure 2.11, and we chose to only generate those with more than 300 input photos.

To compute the final time-lapses, we subsampled time-lapse candidate locations containing more than 1000 photos, by choosing the 1000 closest images under our camera selection criteria. We generated time-lapse videos at a resolution of 1200 pixels in its larger dimension. Figure 2.12 shows the effects of the temporal regularization weight λ on the output time-lapse videos. We found $\lambda = 100$ to be a good compromise between preserving high frequency details and removing flickering in the output sequence. We set the scale parameter of the Huber loss in Equation 2.3 to $4/255$ for the data term, i.e., 4 pixel values, and to $1/255$ for the temporal term. We use Ceres Solver [4] to solve for the temporal appearance independently



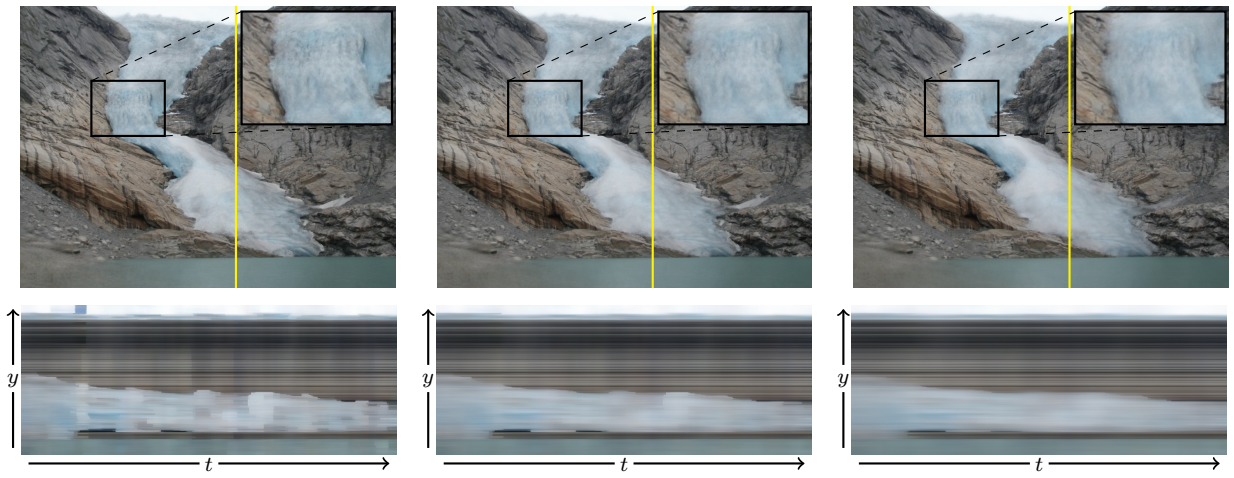
(a) 100 photos (b) 200 photos (c) 400 photos (d) 800 photos

Figure 2.11: Frames of the Goldman Sachs Tower time-lapse video computed with different amount of input images, shown at one third (top row) and two thirds (bottom row) of the sequence. Using fewer input photos produces spurious edges and flickering in the sky region, and the different stages of construction only become clearly visible with more than 400 input photos.

per color channel. Although our depthmap parameter choices worked reasonably well for most scenes, we fine-tuned the depthmap estimation parameters for some of the sequences in the video, in particular, the size of the NCC filter and the weight of the spatial term.

For efficiency, we computed depthmaps at lower spatial (800 pixels) and temporal (500 images) resolution. To generate final time-lapse videos, we play back the regularized output frames at a rate of 120 frames per second (subsampled by 4x to achieve 30fps), meaning that time proceeds at a rate proportional to the rate of photos taken.

A typical time-lapse with 1000 input posed photos takes about 6 hours to compute on a single machine, split equally between viewpoint and appearance stabilization. SfM reconstruction of 1000 photos takes 16 hours for matching and 1 hour for reconstruction with VisualSFM [117]. While the algorithms can be optimized a lot more for efficiency, we point out that a few hours is negligible compared to the time period of several years it took



(a) $\lambda = 25$ (b) $\lambda = 50$ (c) $\lambda = 100$

Figure 2.12: Effects of the temporal regularization weight λ. Top row: resulting frames using different values of λ for the Briksdalsbreen Glacier scene. Note that high frequency detail on the moving glacier is lost with increasing λ. Bottom row: y-t profiles of the pixel column highlighted yellow, showing how smaller λ values create flickering artifacts in the sequence, seen as vertical lines in the temporal profile.

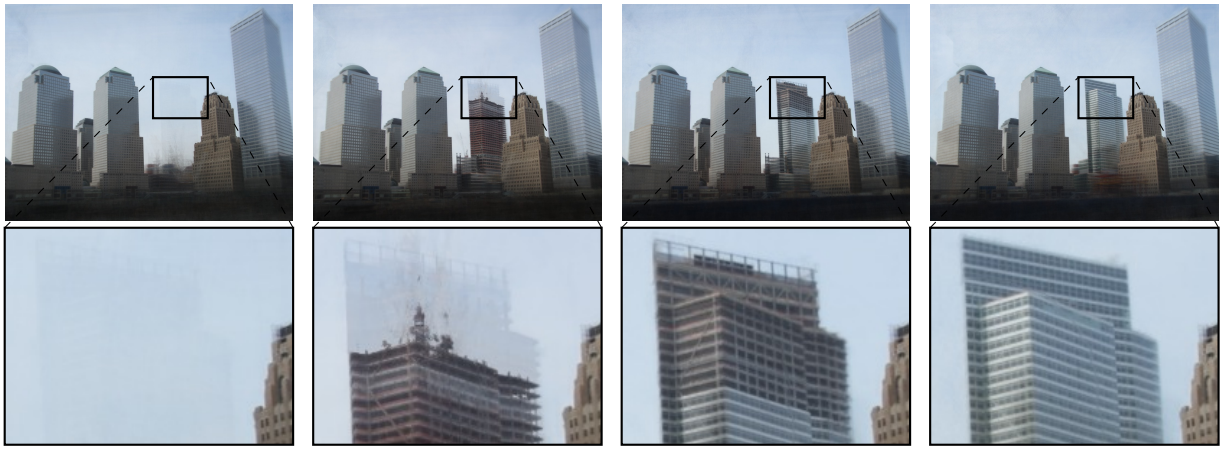
to capture the photos.

For the special case of Briksdalsbreen Glacier, we expanded the time-lapse with more online photos, as our sequence contained few recent photos. We downloaded images from Flickr using a manually specified query, e.g., "Briksdalsbreen Glacier", and added them to the reconstruction using 2D-to-3D matching techniques to register the images.

The resulting time-lapse videos are shown in the supplemental video[1]. They cover a broad range of interesting transformations:

- **Construction**: from individual buildings to whole skylines. The time-lapse of the Goldman Sachs Tower (Figure 2.13), shows the building rise from the ground, followed by windows coming in.

[1]The supplemental video is available in the project website:
`http://grail.cs.washington.edu/projects/timelapse/`



(a) Goldman Sachs Tower, New York City, USA

Figure 2.13: Selected frames of the construction phases of the Goldman Sachs Tower, in New York City. Note how first the building's steel frame is constructed, and then windows are installed floor by floor on the skyscraper.

- **Changing cities**: smaller changes in the appearance of cities, like billboards or changes in urban elements, like sidewalks, etc. (see video).
- **Vegetation**: plants and trees growing, like the trees in the Butchart Gardens (see video).
- **Waterfalls**: we found that waterfalls are constantly changing, as branches dry up and new ones appear (Figure 2.14(a)).
- **Renovations**: monuments being renovated, like the Basilica of St. Maria of Salute in Venice, in Figure 2.14(b).
- **Seasons**: seasonal changes, like the blooming cycles of the flowers in Lombard Street (Figure 2.14(c)).
- **Geological changes**: retreating glaciers, erosion or, like in Figure 2.14(d), the growth of a hot spring in Yellowstone due to the deposit of minerals.
- **Stationary**: some scenes are interesting because of how little they change. For example, the Swiss Guard is so still, that it becomes part of the time-lapse of an entrance

(a) Galovac Waterfall, Plitvice Lakes, Croatia

(b) St. Maria of Salute, Venice

(c) Lombard Street, San Francisco, USA

(d) Mammoth Hot Springs, Yellowstone, USA

Figure 2.14: Selected frames of mined time-lapses, showing different phenomena captured. (a) New branch appears in Galovac Waterfall. (b) Renovation of the St. Maria of Salute Basilica. (c) Blooming of flowers in Lombard Street. (d) Hot spring terraces in Yellowstone grow and change color due to the deposition of minerals.

to the Vatican (see video).

We evaluated a random subset of 500 time-lapses for 1) reconstruction quality, rating them as "good" or "bad", and 2) interestingness, either interesting or not interesting. We found about 45% of the discovered time-lapses to be both good and interesting, 14% only good and 25% only interesting.

2.6.1 Failure modes

We observed a number of interesting failure modes in our system. As noted by 4D Cities, timestamps of online photos are not accurate. When many photos are incorrectly timestamped, our regularization approach can generate spurious halos, like in the second inset of the Goldman Sachs Tower (Figure 2.13).

The time-lapse of Las Vegas (see video), shows blurring in areas where the geometry changes significantly over time. These artifacts can be eliminated when recovering a time-varying depthmap of the scene, as we show in the next chapter.

In other cases, the 3D reconstruction (SfM or stereo) fails. For example, in the Mendelhall Glacier scene (see video), some cameras are registered to features on the moving glacier and our time-lapse video fails to stabilize the background. Such scenes pose a special challenge, as they break the assumptions in Structure-from-Motion systems.

Another limitation of our system are scenes whose recovered 3D models contain both day and night photos. The synthesized time-lapses show an unrealistic "twilight" effect that averages the day and night photos and flickers over time, as seen in the Hong Kong skyline time-lapse (see video).

Our depthmaps are inaccurate in regions that are known to be challenging for stereo algorithms, such as oblique surfaces, like ground planes, clutter or occlusions, like busy squares, or thin structures.

Addressing these limitations is a great topic for future work.

2.7 Conclusion

In this chapter, we introduced an approach to mine time-lapses from Internet photos, that reveals the world's visual history in the Internet photo sharing era. Our system discovered 10,728 time-lapses that show how the world's most popular landmarks are changing over time. Our method stabilizes the time-lapse video sequence so that the underlying changes in the scene become visible. The depicted changes include buildings under construction, glaciers retreating, plants growing, seasonal changes, and many geological processes.

The scale and ubiquity of our mined time-lapses creates a new paradigm for visualizing global changes. As more photos become available online, mined time-lapses will visualize even longer time periods, showing more drastic changes.

Chapter 3

3D TIME-LAPSE RECONSTRUCTION FROM INTERNET PHOTOS

The time-lapse videos presented in the previous chapter reveal many fascinatings changes in the world's most famous landmarks. However, professional photographers exploit small camera motions to capture even more engaging time-lapse sequences [67]. By placing the camera on a controlled slider platform the captured sequences show compelling parallax effects, that add depth and dynamism to the scene. In this chapter, we present an extension to our previous method to mimic these effects, by synthesizing 3D time-lapse video sequences, where the virtual camera moves continuously both space and time.

We introduce key new generalizations that account for time-varying geometry and enable virtual camera motions. Given a user-defined camera path through space and over time, we first compute time-varying depthmaps for the frames of the output sequence. Using the depthmaps, we compute correspondences across the image sequence (aka. "3D tracks"). We then regularize the appearance of each track over time (its "color profile"). Finally, we reconstruct the time-lapse video frames from the projected color profiles.

The key contributions of this chapter are the following: 1) recovering time-varying, temporally consistent depthmaps from Internet photos via a more robust adaption of [121], 2) a 3D time-lapse reconstruction method that solves for the temporal color profiles of 3D tracks, and 3) an image reconstruction method that computes hole-free output frames from projected 3D color profiles. Together, these contributions allow our system to correctly handle changes in geometry and camera position, yielding more compelling time-lapse results compared to the static camera ones.

This chapter describes work that was originally published in the 2015 International Conference on Computer Vision [74].

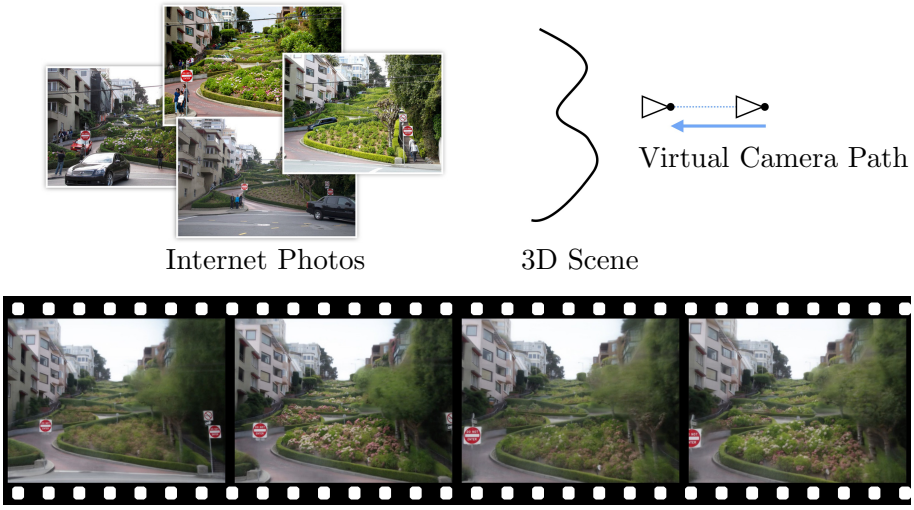


Figure 3.1: In this chapter we introduce a technique to produce high quality 3D time-lapse movies from Internet photos, where a virtual camera moves continuously in space during a time span of several years. Top-left: Sample input photos of the gardens in Lombard Street, San Francisco. Top-right: Schematic of the 3D scene and the virtual camera path. Bottom: Example frames of the synthesized 3D time-lapse video.

3.1 Related Work

The approach described in the previous chapter assumes a static scene and recovers one depthmap that is used to warp the input images into a static virtual camera. A temporal regularization over individual pixels of the output volume recovers a smooth appearance for the whole sequence. The static scene assumption proved to be a failure mode of that approach resulting in blurring artifacts when scene geometry changes. We address this problem by solving for time-varying geometry, and extend the appearance regularization to 3D tracks and moving camera paths.

Parallax Photography, by Zhang *et al.* [123], creates content-aware camera paths that optimize for parallax effects in carefully collected datasets. Additionally, Snavely *et al.* [104]

discover orbit paths that are used to navigate Internet photo collections more efficiently. In our system, the user specifies the camera path as input.

Kopf *et al.* [64] generate smooth hyper-lapse videos from first-person footage. Their technique recovers scene geometry to stabilize the video sequence, synthesizing views along a smoothed virtual camera path that allows for faster playback.

Although multi-view stereo has been an active topic of research for many years [96], few works have looked into time-varying reconstruction outside of carefully calibrated datasets. Zhang *et al.* [122] reconstruct time-varying depthmaps of moving objects with a spacetime matching term. Larsen *et al.* [70] compute temporally consistent depthmaps given calibrated cameras using optical flow to enforce depth consistency across frames. Zhang *et al.* [121] introduce a method to recover depthmaps of a static scene from handheld captured video sequences. Their method first computes a 3D pose for every frame, and then jointly optimizes the depthmaps for every frame, using a temporal consistency term. We extend their approach to handle dynamic scenes, and adapting it to Internet photo collections.

Additionally, Klose *et al.* [62] propose a novel sampling based scene-space video processing framework that first computes a rough depthmap per frame of the video, then gathers several temporal 3D samples per pixel and computes various video processing operations on the samples like denoising or deblurring. Our approach works instead on unstructured photo collections but resembles their framework in that we leverage the 3D structure of the scene to compute pixel colors in the generated time-lapses by regularizing the temporal color profiles of 3D tracks.

3.2 Overview

Given an Internet photo collection of a landmark, we seek to compute time-lapse video sequences where a virtual camera moves continuously in time and space. As a preprocessing step, we compute the 3D pose of the input photo collection using Structure-from-Motion (SfM) techniques [3].

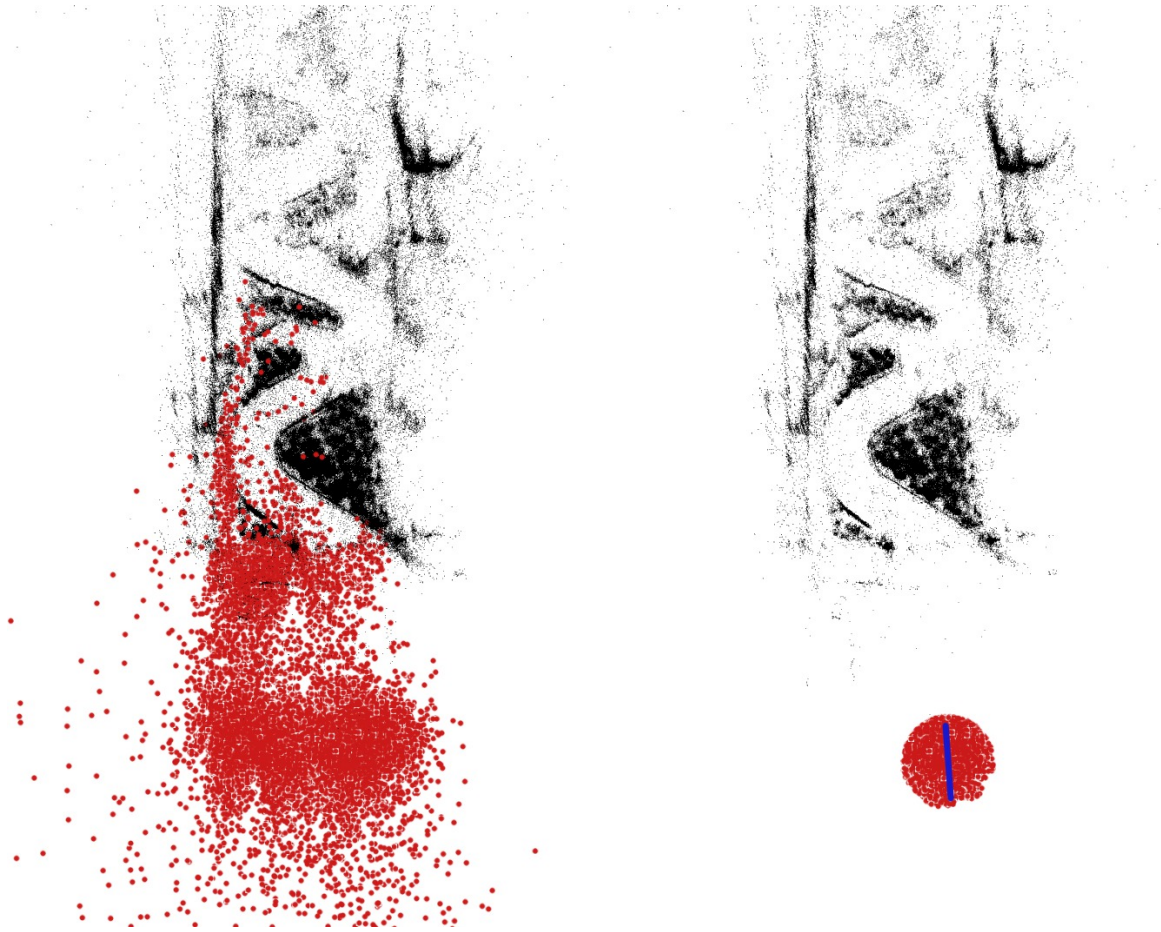
First, a user specifies a desired virtual camera path through the reconstructed scene.

The user starts by choosing a reference camera in the scene. Good reference cameras can be obtained using the scene summarization approach of [102], as described in the previous chapter. The user then selects a parameterized motion path type, such as an orbit around a 3D point or a "push" or "pull" motion path [67]. Finally, the user chooses the motion's length along the chosen path with respect to the 3D reconstruction scale. To help the user in this process, the system previews the chosen camera motion path by computing a depthmap for the reference camera and the median color of the temporal sequence, and rendering this simple approximate geometry and texture over the motion path. Automating the motion length selection is challenging, as the best results depend on the pixel velocities, the amount of change present in the scene, the quality of the 3D reconstruction, and the desired aesthetic of the output time-lapse.

To define an orbit motion path, we first define the center of the scene c as the 3D point that lies in the optical axis of the reference image and whose depth is the average depth of the SfM tracks visible in the reference camera, discounting the closest and farthest 0.5% of the tracks. The virtual camera for every frame is then determined by first displacing the camera center by a number of pixels per frame along its horizontal axis and then rotating the camera along its vertical axis so the projection of the center of the scene c stays fixed. To define a "push" or "pull" camera move, the camera is moved along its optical axis by a certain amount every frame.

Our system starts by computing time-varying, temporally consistent depthmaps for all output frames in the sequence, as described in Section 3.3. Section 3.4 introduces our novel 3D time-lapse reconstruction, that computes time-varying, regularized color profiles for 3D tracks in the scene. We then present a method to reconstruct output video frames from the projected color profiles. Finally, implementation details are described in Section 3.5 and results are shown in Section 3.6.

In the following, we only consider images whose cameras in the 3D reconstruction are close to the reference camera. We use the image selection criteria described in Section 2.3, that selects cameras by comparing their optical axis and camera center to those of the reference



(a) SfM Reconstruction (b) Selected cameras and motion path

Figure 3.2: Illustration of the camera selection process and camera path computation for the Lombard Street scene. Left: Top-down view of the SfM reconstruction, where the 3D points are shown in black and the recovered camera centers are represented as red points. Right: Same top-down view showing in red the camera centers of the selected cameras used for the time-lapse reconstruction, and in blue the camera centers of the virtual camera frames of the time-lapse, that uses a "push" camera motion.

camera.

Figure 3.2 visualizes the SfM reconstruction of the Lombard Street scene together with the selected cameras for the time-lapse reconstruction and the chosen camera path, a "push" camera move. Note that in this case, the length of the camera motion is close to the diameter of the set of selected cameras in the scene. However, these do not have to coincide and it is often the case that the preferred camera motion is much smaller compared to the extent of the selected cameras.

We use the same terminology as in the previous chapter. Each photo in the input collection consists of an image I_i, a registered camera C_i and a timestamp t_i. We also define the sequence $\mathcal{I} = (I_1, \ldots, I_N)$ as the chronologically sorted input image sequence. The output 3D time-lapse sequence is composed of M output frames whose views V_j are equally spaced along the virtual camera path and span the temporal extent of the input sequence, from earliest to the latest photo.

3.3 Time-Varying Depthmap Computation

In this section we describe how to compute a temporally consistent depthmap for every view in the output sequence. The world changes in different ways over time spans of years compared to time spans of seconds. In multi-year time scales, geometry changes by adding or substracting surfaces, like buildings being constructed or plants growing taller, and we design our algorithm to account for such changes.

Recovering geometry from Internet photos is challenging, as these photos are captured with different cameras, different lighting conditions, and with many occluders. A further complication is that included timestamps are often wrong, as noted in previous work [47, 77]. Finally, most interesting scenes undergo changes in both texture and geometry, further complicating depthmap reconstruction.

3.3.1 Problem Formulation

Our depth estimation formulation is similar to that of [121], except that we 1) use a Huber norm for the temporal consistency term to make it robust to abrupt changes in geometry, and 2) replace the photo-consistency term with the one described in Section 2.4 which is also robust to temporally varying geometry and appearance changes which abound in Internet photo collections.

We pose the problem as solving for a depthmap D_j for each synthesized view V_j, by minimizing the following energy function:

$$\sum_j \left[E^d(D_j) + \alpha E^s(D_j)\right] + \sum_{j,j'} \beta_{j,j'} E^t(D_j, D_{j'}) \tag{3.1}$$

where E^d is a data term based on a matching cost volume, E^s is a spatial regularization term between neighboring pixels, and E^t is a binary temporal consistency term that enforces the projection of a neighboring depthmap $D_{j'}$ into the view V_j to be consistent with D_j. The binary weight $\beta_{j,j'}$ is non-zero only for close values of j and j'.

Given the projected depthmap $D_{j' \to j}$ of the depthmap $D_{j'}$ into view V_j, we define the temporal regularization term for a pixel p in V_j as:

$$E^t(D_j, D_{j'})(p) = \delta\left(D_j(p) - D_{j' \to j}(p)\right) \tag{3.2}$$

if there is a valid projection of $D_{j'}$ in view V_j at p and 0 otherwise, and where δ is a regularization loss. We use z-buffering to project the depthmap so that the constraint is enforced only on the visible pixels of view V_j. Zhang *et al.* [121] assume a Gaussian prior on the depth of the rendered depthmap, equivalent to δ being the L_2 norm. In contrast, our scenes are not static and present abrupt changes in depth, that we account for by using a robust loss, the Huber norm.

The data term $E^d(D_j)$ is defined as the matching cost computed from a subset of input photos $\mathcal{S}_j \subset \mathcal{I}$ for each view V_j. We choose the subset as the subsequence of length $l = 0.15N$ centered at the corresponding view timestamp.

Using the images in subset $\mathcal{S}_j$, we compute aggregate matching costs following Section 2.4, that we briefly review here. First, we generate a set of fronto-parallel planes to the view V_j using the computed 3D SfM reconstruction. We set the range to cover all but the 0.5% nearest and farthest SfM 3D points from the camera. In scenes with little parallax this approach might still fail, so we further discard SfM points that have a triangulation angle of less than 2 degrees.

For each pixel p in view V_j and depth d, we compute the pairwise matching cost $C^j_{a,b}(p,d)$ for images $I_a, I_b \in \mathcal{S}_j$, by projecting both images onto the fronto-parallel plane at depth d and computing normalized cross correlation with filter size 3×3. We adapt the best-k strategy described in [56] to the pairwise matchings costs and define the aggregated cost as:

$$C^j(p,d) = \mathbf{median}_{a \in \mathcal{S}_j} \left(\mathbf{median}_{b \in \mathcal{S}_j} C^j_{a,b}(p,d) \right) \tag{3.3}$$

Finally, the spatial regularization E^s consists of the differences of depth between 4 pixel neighborhoods, using the Huber norm, with a small scale parameter to avoid the stair-casing effects observed by [81].

3.3.2 Optimization

The problem formulation of Equation 3.1 is hard to solve directly, as the binary temporal regularization term ties the depth of pixels across epipolar lines. We optimize this formulation similarly to [121], by first computing each depthmap D_j independently, *i.e.*, without the consistency term E^t, and then performing coordinate descent, where the depthmap D_j is optimized while the others are held constant. We iterate the coordinate descent through all depthmaps for two iterations, as the solution converges quickly.

We solve the problem in the continuous domain with non-linear optimization [4], adapting the data term to the continuous case by interpolating the cost values for a pixel at different depths using cubic splines. We initialize each individual depthmap D_j by solving the MRF formulation of Section 2.4 for its corresponding support image set $\mathcal{S}_j$.

The joint optimization produces more stable depthmaps that exhibit fewer artifacts than

(a) Sample input photos (b) Initialized depthmap (c) After joint optimization

Figure 3.3: Results of our time-varying depthmap reconstruction. a) Sample input photos at different times from the Las Vegas skyline scene (not aligned to virtual camera). b) Initialized depthmap for the corresponding time of the photos on the left. c) Jointly optimized depthmaps. Note that artifacts near the top in the second depthmap are fixed after the joint optimization. The improvements to temporal consistency are dramatic and better seen in the supplementary video. Photo credits: Butterbean and Alex Proimos.

the initialized ones without the temporal consistency term. Figure 3.3 shows examples of recovered time-varying depthmaps.

3.4 3D Time-Lapse Reconstruction

Our goal is to produce photorealistic time-lapse videos that visualize the changes in the scene while moving along a virtual camera path. We pose the 3D time-lapse reconstruction problem as recovering time-varying, regularized color profiles for 3D tracks in the scene. A 3D track is a generalization of an image-to-image feature correspondence, which accounts

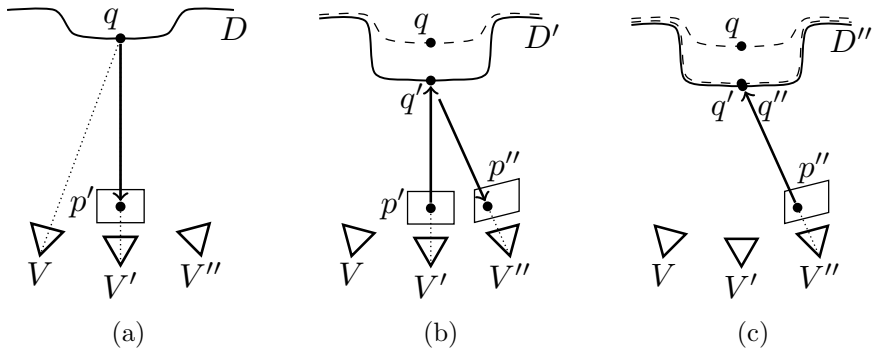


Figure 3.4: Diagram of how a 3D track is generated in three consecutive views. a) A 3D point q visible in view V is projected to view V' at pixel p'. b) Pixel p' is backprojected onto the depthmap D', creating the 3D point q'. Then, the 3D point q' is projected into view V'' at pixel p''. c) Finally, pixel p'' is backprojected onto the depthmap D'', creating the last point in the track q''. The computed track is $t = (q, q', q'')$. Note that because the geometry remains unchanged between V' and V'', the points q' and q'' are the same.

for changes in 3D scene structure, and occlusions between views (See Fig. 3.4). First, we generate 3D tracks by following correspondences induced by the depthmap and the camera motion. We then solve for the temporal appearance of each 3D track, by projecting them onto the corresponding input images and solving for time-varying, regularized color profiles. Finally, we reconstruct the output time-lapse video from the projected color profiles of the 3D tracks.

3.4.1 Generating 3D Tracks

We generate 3D tracks that follow the flow induced in the output sequence by the time-varying depthmap and the camera motion. Ideally, a track represents a single 3D point in the scene, whose appearance we want to estimate. However, occlusions and geometry changes may cause a track to cover multiple 3D points. Since the appearance regularization described in the next subsection is robust to abrupt changes in appearance, our approach

works well even with occlusions.

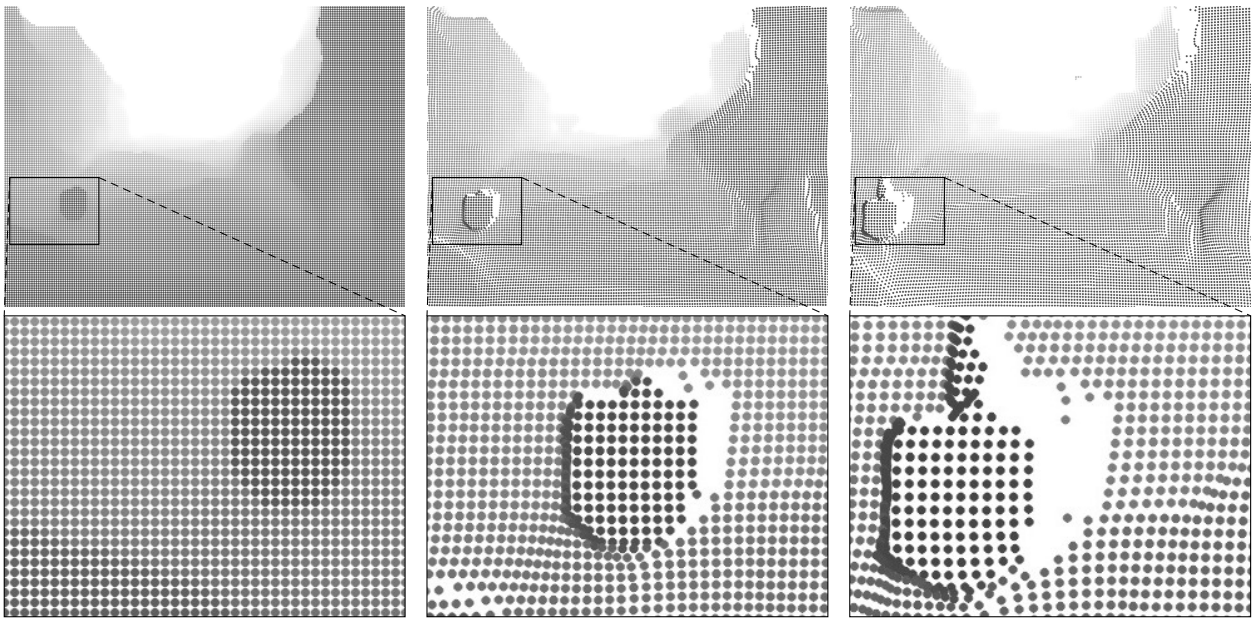
A 3D track is defined by a sequence of 3D points $t = (q_{j_1}, \ldots, q_{j_n})$ for corresponding output views $j_1, \ldots, j_n$. To generate a 3D track, we define first a 3D point q for a view V that lies on the corresponding depthmap D. Let p' be the projection of the 3D point q onto the next view V'. We then define the track's next 3D point q' as the backprojection of pixel p' onto the corresponding depthmap D'. We compute the next 3D point q'' by repeating this process from q'. We define a whole track by iterating forwards and backwards in the sequence, and we stop the track if the projection falls outside the current view. 3D tracks are generated so that the output views are covered with sufficient density as described in Section 3.4.3.

Figure 3.4 shows the 3D track generation process. Note that when the geometry is static, points in a 3D track remain constant thanks to the robust norm used in the temporal consistency term, that promotes depthmap projections to match between frames. While drift can occur through this chaining process, in practice this does not affect the quality of the final visualizations.

Figure 3.5 shows the movement of 3D tracks in the Lombard Street scene. Note how the 3D tracks accumulate at occluding edges, because in our formulation the 3D tracks are continued in the case of occlusion, and start tracking the occluding surface. Also note holes appearing when parts of the scene become disoccluded, like on the right of the street sign. These holes are not a problem as we can compute more tracks in those areas to model the scene densely.

3.4.2 Regularizing Color Profiles

We want to recover a time-varying, regularized color profile for each 3D track t. This is challenging as Internet photos display a lot of variation in appearance and often contain outliers, as noted in Section 3.3. We make the observation that the albedo of most surfaces in the real world does not change rapidly, and its variability in appearance stems mostly from illumination effects. Intuitively, we would like our time-lapse sequences to reveal the



(a) Frame 1 (b) Frame 75 (c) Frame 150

Figure 3.5: Visualization of how 3D tracks move throughout the frame in the Lombard Street scene. Left: A set of 3D tracks is initialized in a grid pattern in the first frame of the sequence and are represented as colored circles with their colors corresponding to their depths. Middle and right: The 3D tracks are continued forward in time and are shown at frame 75 and frame 150 of 200. Note how the tracks accumulate at the left edge of the street sign, as they become occluded by the sign and start tracking the occluding surface. In contrast, on the right side of the street sign the background becomes disoccluded and creates a hole in the representation, as no tracks were initialized on the background surface.

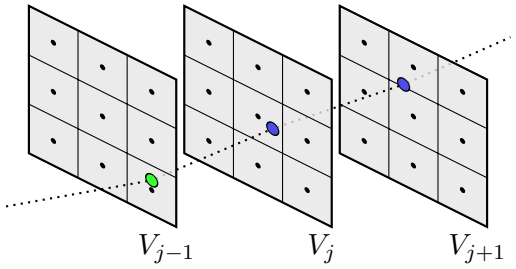


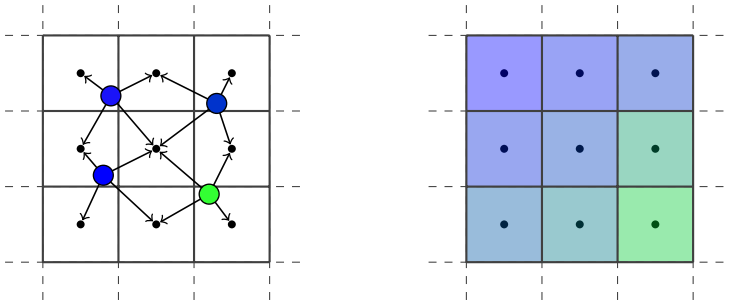
Figure 3.6: Projected temporal color profiles of a 3D track t into three views. The views are represented by a pixel grid, with the pixel centers marked as black dots. The projected temporal color profiles are defined by a real-valued projected position p_j^t into view j and a time-varying, regularized color y_j^t. The projected profile is shown as a sequence of colored circles, projected on each view, linked by a dashed line.

infrequent texture changes (the signal) while hiding the variability and outliers of the input photo collection (the noise).

To solve for time-varying color profiles, Section 2.5 proposed a temporal regularization term with a robust norm, that recovers piecewise continuous appearances of pixels in an output image sequence. Here, we adapt the approach to regularize the color profile of 3D tracks instead of individual pixels in the output sequence, to allow for the virtual camera to move in space.

Given a 3D track $t = (q_{j_1}, \ldots, q_{j_n})$, we define its appearance in view V_j as the RGB value $y_j^t \in [0,1]^3$. To compute y_j^t, we first assign input images to their closest view in time and denote these images assigned to view V_j by the support set $\mathcal{S}'_j \subset \mathcal{I}$. Note that the sets $\mathcal{S}'_j$ are not overlapping, whereas the support sets $\mathcal{S}_j$ used for depthmap computation are. We then project the 3D point q_j to camera C_i using a z-buffer with the depthmap D_j to check for occlusions and define x_i^t as the RGB value of image i at the projection of q_j.

We obtain a time-varying, regularized color profile for each 3D track t by minimizing the



(a) Projected color profiles (b) Reconstructed image

Figure 3.7: Visualization of the output frame reconstruction algorithm from projected color profiles. Left: Projected color profiles at a given view shown as colored dots in the output frame with their bilinear interpolation weights shown as arrows from the projected sample to pixel centers. Right: We reconstruct an image that minimizes the bilinear interpolation residuals of the projected color profiles.

following energy function:

$$\sum_j \sum_{i \in \mathcal{S}'_j} \delta_d \left(\left\| x_i^t - y_j^t \right\| \right) + \lambda \sum_j \delta_t \left(\left\| y_{j+1}^t - y_j^t \right\| \right) \tag{3.4}$$

where the weight λ controls the amount of regularization, and both δ_d and δ_t are the Huber norm, to reduce the effects of outliers in x_j^t and promote sparse temporal changes in the color profile.

In contrast to the previous chapter, the color profiles of the 3D tracks do not correspond to pixels in the output frames. We thus save the color profile y^t, together with the 2D projections p_j^t of the track 3D points q_j^t into the view j, as *projected profiles* that are used to reconstruct the output frames. Figure 3.6 shows a diagram of a projected color profile.

3.4.3 Reconstructing Video from Projected Profiles

Given regularized projected color profiles computed for a set of 3D tracks $\mathcal{T}$, we seek to reconstruct output frames of the time-lapse video that best fit the recovered color profiles.

We cast the problem of reconstructing each individual frame as solving for the image that best matches the color values of the projected color profiles when applying bilinear interpolation at the profiles' 2D projections. Figure 3.7 visualizes the reconstruction process, where the output pixels' color values are related to the projected profiles' samples by bilinear interpolation weights.

For a given output view V_j, let $Y_{u,v} \in [0,1]^3$ be the RGB value of the pixel $(u,v) \in \mathbb{N}^2$ in the synthesized output frame Y. Let the regularized projected profile for a track t at view V_j have an RGB value y^t and a 2D projection $p^t \in \mathbb{R}^2$. We solve for the image Y that minimizes

$$\sum_{t \in \mathcal{T}} \left\| y^t - \textstyle\sum_{s=1}^{4} w_s^t Y_{u_s^t, v_s^t} \right\|_2 \tag{3.5}$$

where u_s^t, v_s^t are the integer coordinates of the 4 neighboring pixels to p^t and w_s^t their corresponding bilinear interpolation weights.

The reconstruction problem requires the set of 3D tracks $\mathcal{T}$ to be dense enough that every pixel $Y_{u,v}$ has a non-zero weight in the optimization, *i.e.*, each pixel center is within 1 pixel distance of a projected profile sample. To ensure this, we generate 3D tracks using the following heuristic: we compute 3D tracks for all pixels p in the middle view j of the sequence, so that the 3D track point q_j^t projects to the center of pixel p in V_j. Then, we do the same for all pixels in the first and last frame. Finally, we iterate through all pixels in the output frames Y and generate new 3D tracks if there is no sample within $\epsilon \leq 1$ pixels from the pixel center coordinates.

The reconstruction problem can be badly conditioned, producing artifacts in the reconstructions, such as contiguous pixels with alternating black and white colors. This happens in the border regions of the image that have lower sample density. We avoid such artifacts by using a low threshold value $\epsilon = 0.4$ pixels, so that for each pixel there is a projected profile whose bilinear interpolation weight is > 0.5. Figure 3.8 shows an example frame reconstruction using two different threshold values for ϵ. Using $\epsilon = 0.4$, the output frame pixels in our sequences are covered by an average of four 3D tracks, while foreground pixels

Figure 3.8: Comparison of different values of the 3D track sampling threshold ϵ for the Wall Street Bull scene. Left: Artifacts are visible when $\epsilon = 1$ pixel, with alternating black and white pixels, as the reconstruction problem is badly conditioned. Right: Using $\epsilon = 0.4$ pixel, the artifacts are not present.

on depth discontinuities might be covered by up to hundreds of 3D tracks.

3.5 Implementation

For our system, we use the same photo collections as in the previous chapter. For a single landmark, the 3D reconstructions contain up to 25K photos, and the input sequences filtered with the camera selection criteria in Section 2.3 contain between 500 and 2200 photos. We generate virtual camera paths containing between 100 and 200 frames.

The weights for the depthmap computation are $\alpha = 0.4$ and the temporal binary weight is defined as $\beta_{j,j'} = k_1 \max(1 - |j' - j|/k_2, 0)$ with $k_1 = 30$ and $k_2 = 8$. The scale parameter of the Huber loss used for E^s and E^t is 0.1 disparity values. For appearance regularization, we use the Huber loss for δ_d and δ_t with scale parameter of 1^{-4}, *i.e.*, about 1/4 of a pixel value. Finally, the temporal regularization weight is $\lambda = 25$. We use Ceres Solver [4] to solve for the optimized color profiles, that we solve per color channel independently.

Our multi-threaded CPU implementation runs on a single workstation with 12 cores and 48Gb memory in 4 hours and 10 minutes for a 100 frame sequence. The breakdown is the following: 151 minutes for depthmap initialization, 30 minutes for joint depthmap optimization, 55 minutes for 3D track generation and regularization, and 25 minutes for video reconstruction. For reference, the sequences in the supplemental video contain 200 frames at HD quality (1440×1080) with a depthmap resolution of 640×480 and took about 24 hours to compute. Our execution time is dominated by the cost volume computation for all the views, and we subsample the support sets $\mathcal{S}_j$ to contain at most 100 images without noticeable detrimental effects.

Our generated time-lapses tend to have an overall subtle blur in the whole frame caused by small pixel misalignments in both the SfM reconstruction and our temporal depthmap recovery. This is expected as the precision of both systems is already larger than a pixel. To account for this blur, we apply a sharpening filter as a post-processing step to the output sequences. We found that applying a value of 30% in Adobe's Premiere CS5 is sufficient and does not create visible artifacts.

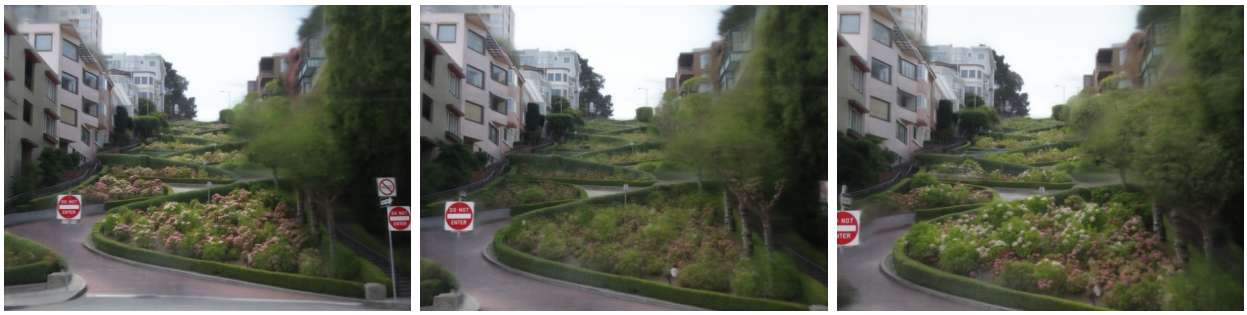
3.6 Results

We generated high-quality 3D time-lapse videos for 14 scenes, spanning time periods between 4 and 10 years. Figures 3.9 and 3.10 show sample frames from six different scenes. The scene of the Charging Bull statue in New York City shows that the statue has moved in the past. This can be seen clearly in Figure 3.11, that shows how the front left hoof slid over the pavement in 2009. We refer the reader to the supplementary video[1] to better appreciate the changes in the scenes and the parallax effects in our 3D time-lapses.

We compare our output frame reconstruction approach with a baseline method that uses splatting of the projected color profiles with Gaussian weights. Each projected profile sample contributes its color to nearby pixels with a weight based on the distance to the pixel center.

[1]The video is available at the project website: `http://grail.cs.washington.edu/projects/timelapse3d/`

(a) Flatiron Building, New York



(b) Lombard Street, San Francisco

(c) Ta Prohm, Cambodia

Figure 3.9: Frames from example 3D time-lapses, with time spans of several years and subtle camera motions. Sequences a) and c) contain an orbit camera path, while b) contains a camera "push". Parallax effects are best seen in the video available at the project website. Limitations of our system include blurry artifacts in the foreground, like in c).

(a) Palette Springs, Yellowstone

(b) Abbey Falls, India

(c) Brikdalsbreen Glacier, Norway

Figure 3.10: Frames from example 3D time-lapses, with time spans of several years and orbit camera motions. Parallax effects are best seen in the video available at the project website. Limitations of our system include blurry artifacts in the foreground, like in a).

Figure 3.11: Two frames of the Charging Bull sequence in New York City that show the statue's movement. Note how the front left hoof slides over the pavement half a cobblestone. This motion is not due to errors in the Structure-from-Motion reconstruction, as the cobblestones are stable throughout the sequence, indicating that the reconstruction is fixed on the pavement and not on the statue.

Figure 3.12 shows that the baseline produces blurred results whereas our approach recovers high frequency details in the output frame.

Figure 3.13 shows a comparison of our 3D time-lapse for the Las Vegas sequence with the result of the approach in the previous chapter, that was noted as a failure case due to changing scene geometry. Our 3D time-lapse result eliminates the blurry artifacts, as the time-varying depthmap recovers the building construction process accurately.

As discussed in Section 3.4.1, our 3D track formulation allows for tracks to jump between surfaces in the case of depth discontinuities or occlusions and relies on the recovery of the temporal color profiles to account for changes in color due to changing surfaces. We ran an experiment where instead we stop 3D tracks at depth discontinuities or occlusions, *i.e.*, we do not continue a track when its depth would change significantly from one frame to another, as measured by the recovered temporal depthmap. We show the results in Figure 3.14. When

Figure 3.12: Comparison of two methods for output frame reconstruction from projected profiles for the Musée D'Orsay scene. Left: baseline method based on Gaussian kernel splatting, with kernel radius $\sigma = 1$. Right: our reconstruction approach. The baseline method produces a blurred reconstruction, whereas the proposed approach recovers high frequency details in the output frame.

using track splitting, the resulting frames contain sharp color edges at depth discontinuities given by recovered time-varying depthmaps. However depth discontinuities that change over time, like the top edge of the skyscraper under construction, are very challenging to reconstruct temporally and any inaccuracy or temporal inconsistency leads to jarring artifacts in the output frames. In contrast, our 3D track formulation is able to reconstruct frames that hide any inaccuracies in the depthmaps and look more realistic compared to a reference image.

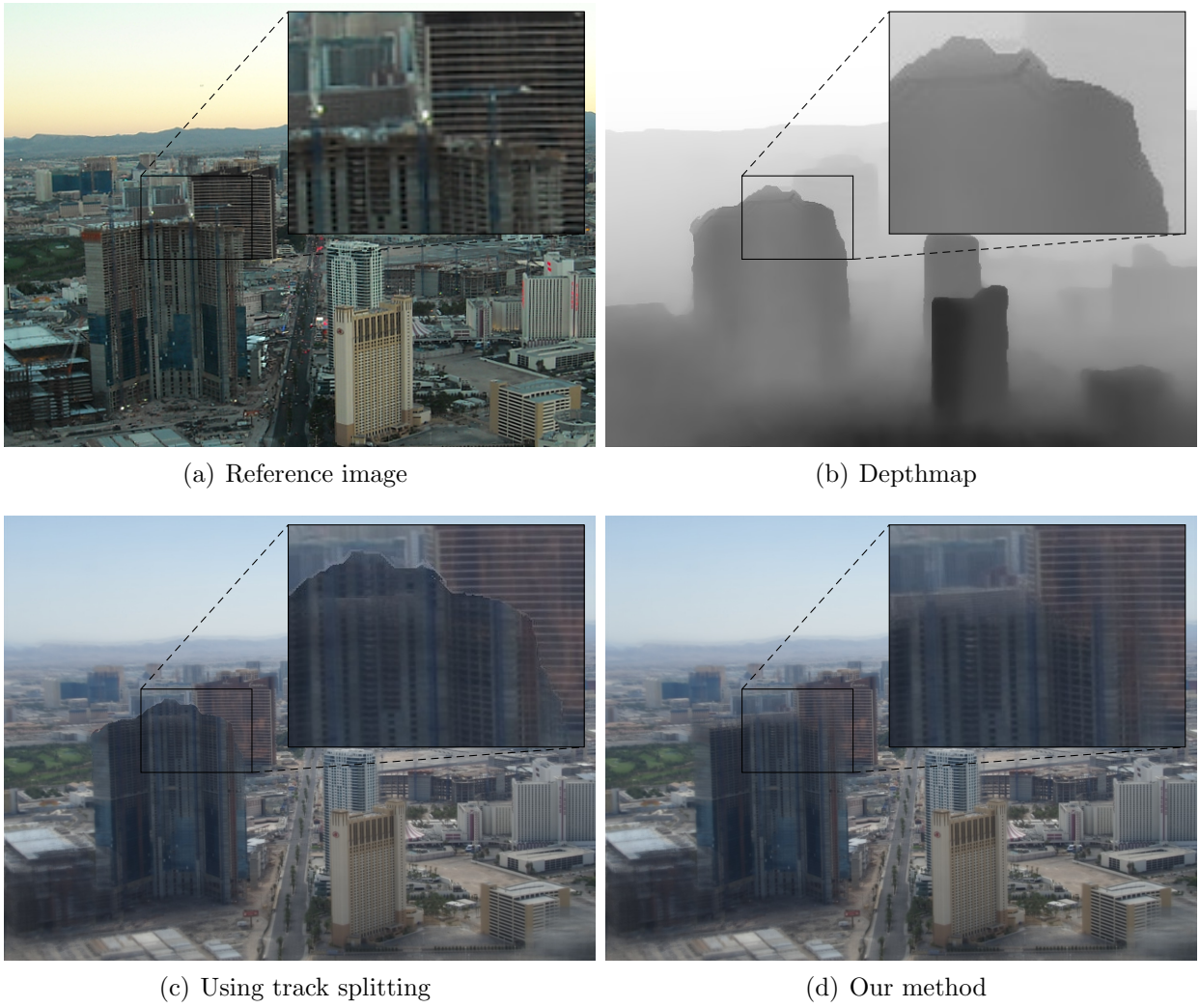
3.6.1 Limitations

We observed a few failure cases in our system.

Thin structures: Our time-varying depthmaps sometimes fail to recover thin structures and our resulting time-lapses blur these thin structures with the background. For example,

(a) Static depthmap (b) Time-varying depthmap

Figure 3.13: Comparison of output time-lapse frames for two different timestamps for the Las Vegas sequence. a) Using a static depthmap solved with a discrete MRF as in [75]. b) Using our time-varying, temporally consistent depthmaps. The static depthmap is not able to stabilize the input images for the whole time-lapse, creating blurry artifacts where the geometry changes significantly. Thanks to the time-varying depthmap, our 3D time-lapses are sharp over the whole sequence.



(a) Reference image

(b) Depthmap

(c) Using track splitting

(d) Our method

Figure 3.14: Comparison of the results when splitting the 3D tracks at depth discontinuities. a) Reference image of around the same time-period. b) Recovered depthmap for the given frame. c) Frame of reconstructed time-lapse using the 3D tracks that stop at depth discontinuities. d) Frame of reconstructed time-lapse where 3D tracks are continued over depth discontinuities. Note that c) shows the discontinuity edge in the depthmap. However, it is not visible in d) and the result looks more visible plausible compared to the reference image. Credits: Creative Commons photo from Flickr user Daniel Ramirez.

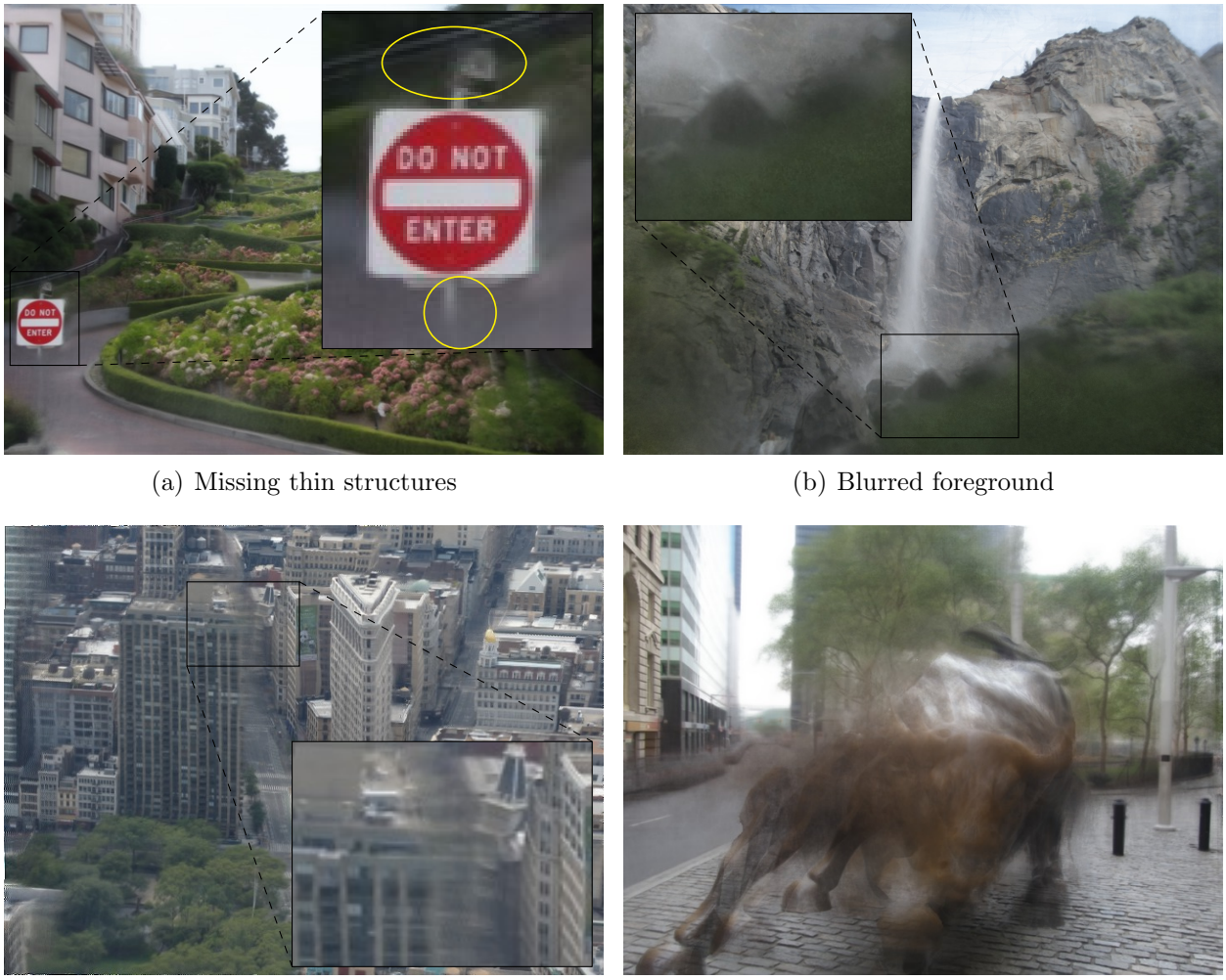
in the Lombard Street sequence in San Francisco the pole of the "Do Not Enter" sign is not reconstructed in the depthmap and the sign appears to be floating, as shown in Figure 3.15(a) Recovering thin structure geometry is an active area of research and is very challenging in the case of unstructured photo collections.

Blurred foreground: In some scenes, the background might be visible through vegetation that is much closer to the camera. The recovered depthmaps fail to recover the vegetation because it exhibits too much parallax and large appearance variations and reconstruct instead the far background. Consequently, in the reconstructed time-lapses the foreground objects bleed into the background and generate blur artifacts with the color of the foreground objects. This can be seen for example in the Bridalveil Falls scene in Yosemite, as shown in Figure 3.15(b), that contains close-up vegetation in front of the far rock wall.

Extrapolation: Our system also generates artifacts when synthesizing viewpoints significantly different than the input photo collection. This happens when a camera looks at a surface not visible in any input photo. For example, in Figure 3.15(c) a view is synthesized for a camera outside the convex hull of reconstructed cameras, showing a face of a building that is not visible from any photo. Future work could consider using visibility information to constrain the virtual camera paths like in [123].

Depthmap temporal resolution: Another limitation of our approach is when the scene geometry changes faster than what our time-varying depthmap can resolve. This happens in the "Charging Bull" statue scene in New York, where the statue changed positions a few times at the beginning of the sequence, leading to a blurred appearance in the first frames of the sequence, as shown in Figure 3.15(d). The limited temporal resolution of our time-varying depthmaps arises from using large temporal windows to compensate for the variability of Internet photos.

Our technique is limited to reconstructing 3D time-lapses given pre-specified camera paths. Future work includes enabling interactive visualizations of these photorealistic 3D time-lapses.



(a) Missing thin structures

(b) Blurred foreground

(c) Extrapolation artifacts

(d) Temporal resolution

Figure 3.15: Examples of failure cases in our system. a) The street sign is not fully reconstructed in the Lombard Street sequence. b) The foreground vegetation in the Bridalveil Falls scene is not reconstructed in the depthmap as it exhibits large amounts of parallax and instead bleeds into the background generating blur artifacts. c) An extended camera orbit contains a virtual camera far from the set of input cameras causing blurry artifacts at occlusion boundaries in the Flatiron Building dataset. d) The "Charging Bull" statue changes position more often than the temporal resolution of our time-varying depthmap, that is unable to stabilize the sequence and leads to blurring.

3.7 Conclusion

In this chapter we introduce a method to reconstruct 3D time-lapse videos from Internet photos where a virtual camera moves continuously in time and space. Our method involves solving for time-varying depthmaps, regularizing 3D point color profiles over time, and reconstructing high quality, hole-free output frames. By using cinematographic camera paths, we generate time-lapse videos with compelling parallax effects, that are more appealing than the static camera ones presented in the previous chapter.

Chapter 4

THE 3D WIKIPEDIA

The history of ancient places is not captured in photographs, but in books that describe them. Tourists have long relied on guidebooks and other reference texts to learn about historical sites. While guidebooks are packed with interesting historical facts and descriptions of site-specific objects and spaces, it can be difficult to fully *visualize* the scenes they present. The primary cues come from images provided with the text, but coverage is sparse and it can be difficult to understand the spatial relationships between each image viewpoint. For example, the Berlitz and Lonely Planet guides [14, 37] for Rome each contain just a single photo of the Pantheon, and have a similar lack of photographic coverage of other sites. Even online sites such as Wikipedia, which do not have space restrictions, have similarly sparse and disconnected visual coverage (see Figure 1.3).

Instead of relying exclusively on static images embedded in text, suppose you could create an interactive, photorealistic visualization, where, for example, a Wikipedia page is shown next to a detailed 3D model of the described site. When you select an object (e.g., "Raphael's tomb") in the text, it flies you to the corresponding location in the scene via a smooth, photorealistic transition. Similarly, when you click on an object in the visualization, it highlights the corresponding descriptive text on the Wikipedia page. Our goal is to create such a visualization completely automatically by analyzing the Wikipedia page itself, together with many photos of the site available online (Figure 4.1).

Automatically creating such a visualization presents a formidable challenge. The text and photos, in isolation, provide only very indirect cues about the structure of the scene.

This chapter describes work that was originally published in ACM SIGGRAPH Asia 2013 [91], where I appeared as the second author.

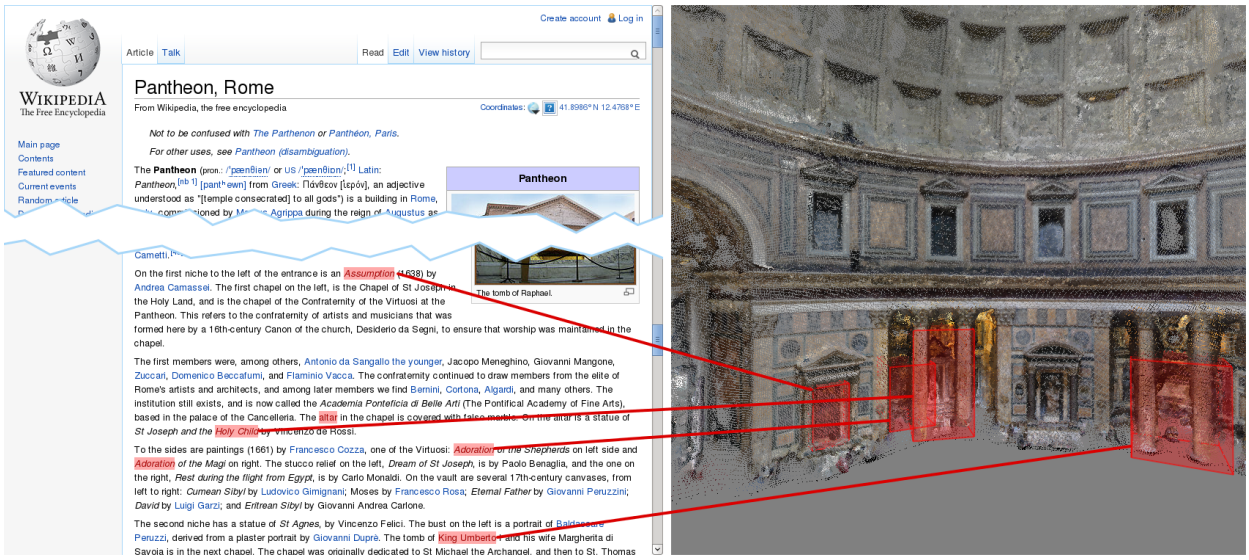


Figure 4.1: Given a reference text describing a specific site, for example the Wikipedia article above for the Pantheon, we automatically create a labeled 3D reconstruction, with objects in the model linked to where they are mentioned in the text. The user interface enables coordinated browsing of the text with the visualization.

Although we can easily gather text describing the world, automatically extracting the names of objects (e.g.,"Raphael's tomb" or "Coronation of the Virgin") is not trivial. For example, we know a noun phrase often describes an entity, which could be an object in the scene. However, it could also name the artist that created the object, or some other unrelated concept. Given the correct names, even more challenging is determining the precise *3D location* of each described object, since most textual descriptions within any given reference text are not accompanied by pictures or other explicit visual cues.

The key to our approach is to mine text and photo co-occurences across all of the Internet. For example, a photo anywhere on the Internet with the caption "Annunciation, Pantheon" signals that it may depict the named fresco. Indeed, a Google image search for "Annunciation, Pantheon" yields perfectly cropped images of the desired object (Figure 4.2,

top). Given a Pantheon reconstruction, these images can be matched directly to the model to label the corresponding regions in 3D. Although this approach allows us to find 3D object locations, our challenge of finding object names in text remains. Our solution is to do a brute-force extraction of *every* noun phrase in the text, execute a Google image search query for that phrase (with ", Pantheon" added at the end), and select only the phrases with images that align with the model. Of course, this simple strategy does not completely solve the problem; image captions and web page co-occurences are notoriously noisy. Searching for correctly named objects can produce multiple matching images (Figure 4.2, middle) and phrases that do not describe actual objects can produce spurious matches (Figure 4.2, bottom). Hence, we treat the image results as a noisy signal to be integrated with other constraints in a joint, learned model for filtering out spurious phrase, image pairs. This approach can be considered as a form of *query expansion* [18, 23, 93] where we issue several queries on pieces of the text and then verify the results.

Our reconstruction and visualization approach is inspired by Photo Tourism [105], and we employ similar techniques to generate 3D models from Flickr photos and to render transitions to photos within those models [116, 117, 118]. Our innovation is not in the rendering per se, but in our ability to automatically transform descriptive texts such as Wikipedia pages into interactive 3D visual experiences, where the text links to corresponding points in a reconstructed 3D model. We show compelling results for several major tourist sites. While no automated method is perfect, we are able to reliably extract many of the objects in each scene, with relatively few errors (we provide a detailed analysis of precision and recall).

4.1 Related work

Our labeling problem lies at the interface between natural language processing and 3D computer vision; a very fertile area with little prior research. An exception is Simon et al.'s work [101] on segmenting and labeling 3D point clouds by analyzing SIFT feature co-occurence in tagged Flickr photos. Their approach works by associating commonly occurring image text tags with the model points contained in the associated images. However, Flickr tags are

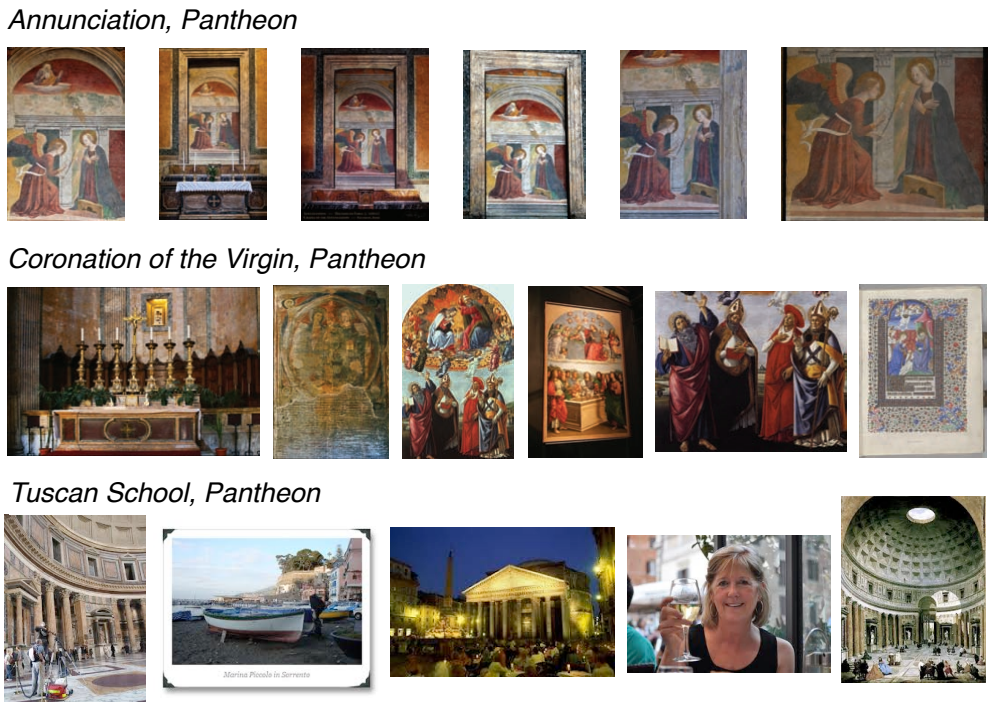


Figure 4.2: The top Google image search results for two objects inside the Pantheon and one distractor string. The reliability of the search results varies. Top row: all returned search results depict the entire or part of The Annunciation. Middle row: Only the second returned search result is correct. Bottom row: An incorrect object description with several images that do depict the Pantheon.

notoriously noisy and far less informative compared to Wikipedia and other authoritative guides. Their approach cannot be applied to Wikipedia, as it requires tagged photos as input.

In the 2D domain, there is a significant literature on correlating regions in images/video to captioned text or keywords, e.g. [10, 25, 69], and on generating sentences or captions for specific images [13, 32, 78]. These approaches reason about a relatively small set of object classes (e.g. *car*, *boat*) via trained object detectors, whereas we reason about object instances (e.g. the *Annunciation*). Furthermore, note that [13] require captioned photographs during

the training of their model. Our use of 3D reconstructions allows us to avoid many of the object detection challenges these approaches face, and limits our system to find only specific instances of objects within the landmark.

Our work builds on recent breakthroughs on reconstructing 3D models of tourist sites from Internet photo collections. These methods are based on structure-from-motion [3, 86, 106] and multi-view stereo [34, 35, 38]. The advent of commodity depth sensors like Kinect has also inspired work in object category recognition in RGB-D and range-scan data [65, 87, 99]. This work is complementary to our effort; we focus on labeling instances.

There is a long history in computer vision on the problem of recognizing images of specific objects or places (instances). Especially relevant is recent work on large-scale image retrieval [23, 83, 103] that operates by matching local features computed at interest points between an input image and a database of labeled images [72]. Also relevant is work that reasons about GPS-tagged images [26, 48]. All of these techniques require a database of labeled objects as reference. In contrast, our focus is to *create* such a database from joint analysis of text and images.

4.2 System Overview

In this chapter, we present a fully automatic system that generates interactive visualizations that link authoritative text sources with photorealistic 3D models. The system requires two types of inputs: one or more reference text sources, such as Wikipedia, and a unique name for the site to reconstruct, such as the Pantheon in Rome.

Figure 4.3 presents an overview of the complete approach. There are two parallel streams of processing. The system downloads a set of images from Flickr by querying for the site name and then automatically reconstructs a 3D model using the freely available VisualSFM package [117], followed by PMVS [35] to generate a dense 3D point cloud. It also does a query expansion analysis of the text, involving image search and registration for each possible noun phrase as described in Section 4.3, to find text that names objects and link it to the reconstructed 3D geometry. Using these object correspondences, our system creates interactive

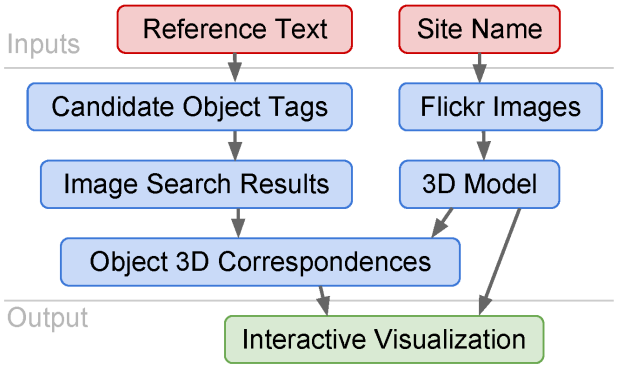


Figure 4.3: System overview.

visualizations, as described in Section 4.4, that emphasize the discovered correspondences, providing innovative navigation experiences for the text and the 3D model.

4.3 Automatic labeling of 3D models from text

In this section, we describe our algorithm for obtaining correspondences between regions on a 3D model to an object tag description in a reference text. Our algorithm consists of three steps: we generate an overcomplete list of candidate object hypothesis from the text; then we obtain their likely location on the 3D model via query expansion; finally we filter the large number of false positive detections by training a classifier over features gleaned from the text and the output of query expansion.

4.3.1 Obtaining object hypotheses from text

For each site, we seek to automatically obtain a list of candidate descriptive phrases. Our texts come from two sources that are freely available online: articles from Wikipedia, and text from other, site specific, third-party web pages. These text sources offer rich descriptions of the site's contents and their spatial layout, along with their history, architectural features, and cultural references.

We use the syntactic structure of the language to define the set of possible descriptive

phrases, primarily leveraging the fact that noun phrases can name physical objects in English. To extract noun phrases, we use the Stanford parser [61], which achieves near state-of-the-art performance and is available as public-domain software. We ran the parser with the default parameter settings. To boost recall, we also extract prepositional phrases that are immediately followed by a noun phrase (e.g. *a fresco of the Annunciation*) and merge adjacent noun phrases (e.g. *a canvas by Clement Maioli of St. Lawrence and St. Agnes*). These additional phrases allow us to overcome parsing errors, e.g., when nouns are incorrectly labeled as prepositions. Extracting them boosts recall and provides a large set of candidates that we will later filter with a joint model that incorporates visual cues. Finally, to reduce false positives, we remove phrases containing only a single stop word, as defined by a commonly used stop word list [107], or only numerals.

4.3.2 From labels to regions via query expansion

Given the automatically obtained list of candidate named objects, we wish to generate proposal regions for their 3D location within the site. We leverage the fact that many objects are photographed in isolation, i.e. with the object shown in whole and filling nearly the full field of view. This *photographer's bias* has been previously used to discover and segment objects within 3D models [101].

For each candidate named object, we search for and download images using Google image search. We construct the query terms by concatenating the extracted noun phrase with the place name (e.g. *central figure Trevi Fountain*). To find candidate regions within the 3D model for the site, we build upon the success of feature matching and geometric verification used to construct 3D models from consumer photographs [106]. We match SIFT key points [72] extracted from the downloaded images to the inlier SIFT key points corresponding to 3D points within the 3D model. Using the putative 2D-3D point correspondences, we recover the camera parameters for the image and inlier 3D points within the 3D model via camera resectioning [46] as shown in Figure 4.4. We find that matching to the 3D model is beneficial for three reasons: (i) our overall goal is to label the 3D model, (ii) we find

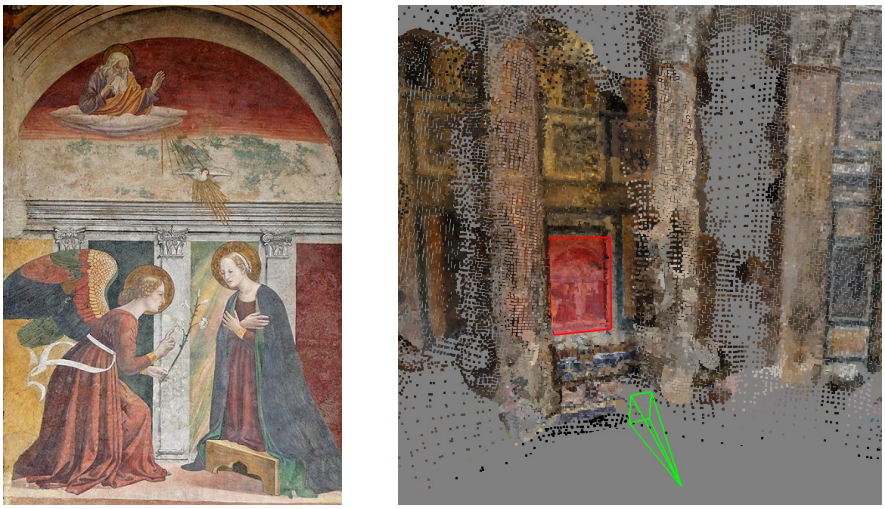


Figure 4.4: Left: image returned from Google image search. Right: section of the 3D model, with a bounding box around the matched 3D and the camera frustrum.

that geometric verification and associating multiple SIFT features to each 3D point offers robustness in the matching step, and (iii) matching to local SIFT keypoints and reasoning about the 3D scene offers robustness to occlusion (c.f. the artwork behind the columns in the Pantheon, which are visible in the 3D model but not in the panorama of Figure 4.6).

As the object of interest is usually depicted within the top set of search results, we perform camera resectioning for the top 6 images returned for each search query. We keep the alignment if camera resectioning finds at least 9 inlier correspondences. We found that verification with at least 9 inlier features almost always yields a correct alignment to the 3D model; using fewer yields incorrect alignments. This requirement discards many images that do not depict the site at all and maintains a high recall for valid images that do depict the site.

4.3.3 Model for filtering hypotheses

The query expansion procedure returns for each candidate object tag a set of 3D points within the 3D model corresponding to candidate locations of the object. While Internet image search returns many valid images for the candidate object tags, there remains a high number of false positives. The false positives often result from generic images of the site that occur often across the different query terms and get successfully aligned to the 3D model. Moreover, we have the additional difficulty of an over-generated list of candidate objects resulting from the output of the natural language processing parser. In this section, we outline a procedure to separate the good object proposals from the bad ones.

Our goal is to extract good object detections from the hypothesis set of object-region pairs. We start by merging highly-overlapping camera frustra corresponding to the aligned images for a given object tag returned from Google image search during the query expansion step. To merge camera frustra, we first project each frustrum onto a reference image (i.e. panorama or perspective photograph) depicting the site that has been registered to the 3D model. We form a bounding box by taking the maximum x, y extent of the projected frustrum. We then merge two frustra if their relative overlap (i.e. ratio of intersection area to their union) exceeds 0.5, with the mean of their bounding boxes returned. This results in a set of object tag and detection frustrum pairs for the site, dubbed the *candidate pool*.

Next, we extract features from the candidate pool and the original text. The visual features include: the number of merged frustra for the candidate; the rank number for the top-ranked image search result that aligned to the 3D model; and the total number of frustra across all object tags that highly overlap the candidate frustrum (a high number indicates a generic viewpoint of the site). The text features include: whether a non-spatial preposition (*ago, as, because of, before, despite, during, for, like, of, since, until*) resides in the same sentence as the extracted noun phrase, which often corresponds to historical descriptions; whether the tag corresponds to an author; and whether an author appears in the same sentence as the tag. We encode the presence of an author as a feature since the authorship

of an object is often described together in the same sentence as the object. We detect the presence of an author in the text by analyzing prepositional *by* dependencies returned from the Stanford parser [61] and return the second string argument in the dependency as the author.

We train a linear classifier $\mathbf{y} = \mathbf{w}^T\mathbf{x}$ over the features $\mathbf{x}$ and their labels $\mathbf{y} \in \{0, 1\}$ using logistic regression across a set of training sites and test on the remaining sites. We use Matlab's `glmfit` function with logit link function. To construct the training set, we project each frustra in the candidate pool for the site onto the reference image and intersect the projected frustra with objects that have been manually labeled via LabelMe [92]. For each labeled object, we keep the object tag/detection frustrum pair that has highest word F-score when comparing the object and labeled tags and having the center of their bounding boxes residing in the other's bouding box. We form the set of positive examples ($\mathbf{y} = 1$) from the tag/frustrum pairs that match to a ground truth label. We form the set of negative examples from tag/frustrum pairs that do not have tag or frustrum overlap with any of the positive training examples. During testing, we perform non-maximum suppression over the detections. We suppress detections if a higher confidence frustrum overlaps a lower confidence one (i.e. their relative overlap exceeds 0.3 and their centers reside in the other's bounding box) or if any of the tag words overlap in the same sentence.

4.4 Visualization tool for browsing objects in online text

We aim to create immersive visualizations that connect information from authoritative text sources to 3D models constructed from Internet photo collections. The extracted correspondences between object tags in the text and regions of the 3D model provide bidirectional links between the two types of media. In this work we present novel ways to explore and navigate these links, providing spatial and contextual information to the text and meaningful descriptions to the 3D model.

Our visualization has two panes: on the left it displays the website containing the reference text, such as Wikipedia, and on the right a 3D visualization of the landmark, that uses

automatically generated 3D bounding boxes to highlight discovered objects. We augment the functionality of the website to enable text-to-3D navigation, 3D-to-text navigation, and automatic tours of the landmarks. Figure 4.5 shows screen captures of our visualizations, but we refer the reader to the supplementary video[1] to best experience the system. In the following subsections, we describe the different navigation modes, as well as the implementation details of the visualization.

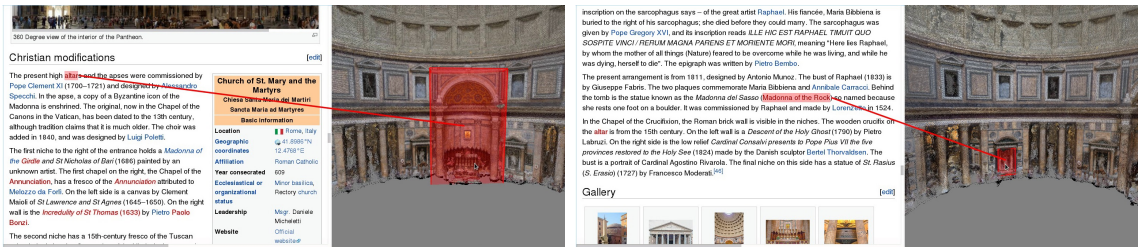
4.4.1 Text-to-3D navigation

In the web pane (left) of our visualization, we create special hyperlinks in the text at all occurrences of discovered object tags. When you mouse over one of the hyperlinks, it first highlights the object tag in the text pane and then the 3D visualization highlights a 3D bounding box around the corresponding object, showing you its location and relative size within the scene. Additionally, to emphasize the connection between the highlighted text and 3D bounding box, the visualization draws a line between them across the two panes.

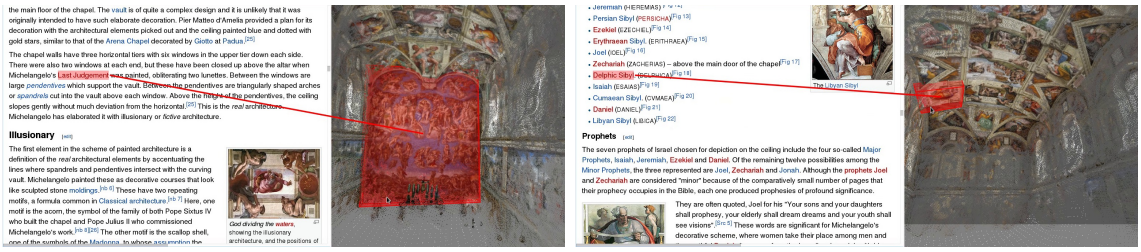
When the named object is not visible in the current viewpoint, the visualization smoothly moves the camera until the object is displayed in the center of the image. To see an image of the highlighted object you can click on the object tag and the visualization first transitions to the viewpoint of a close-up image of the object and then fades in the image. For each object, the visualization chooses the image that maximizes the area of the object's projected 3D bounding box. Once you move the mouse out of the object tag, the line and the bounding box fade out.

The webpages often contain images that depict some of the objects being described. Our visualization further enhances these images by converting them to hyperlinks into the 3D model. When you click on the image, the camera transitions to the corresponding viewpoint in the 3D pane and fades in a high resolution version of it. This functionality is helpful when navigating webpages with many photos (e.g. U.S. Capitol Rotunda Wikipedia page),

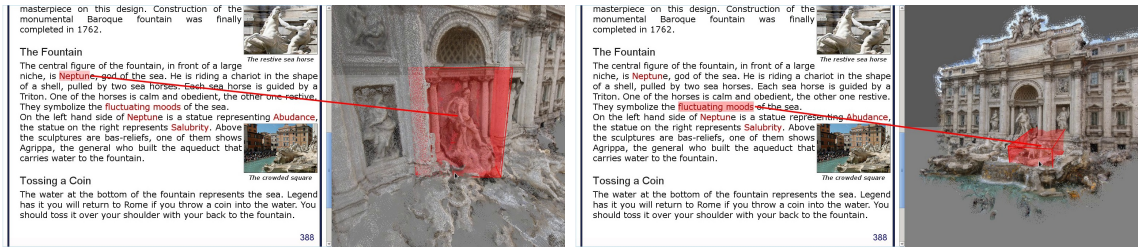
[1]The supplementary video is available in the project website:
`http://grail.cs.washington.edu/projects/label3d/`



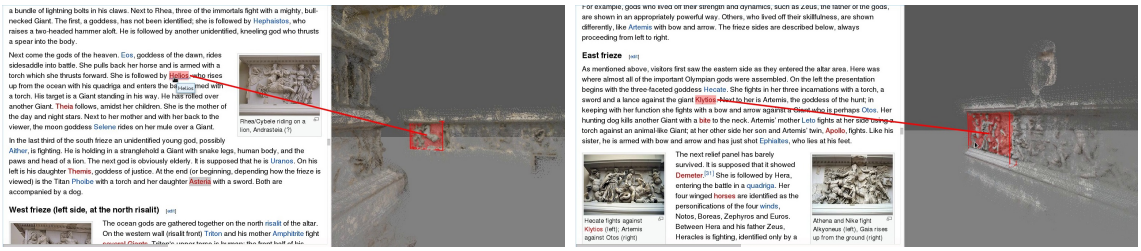
(a) Pantheon, Rome



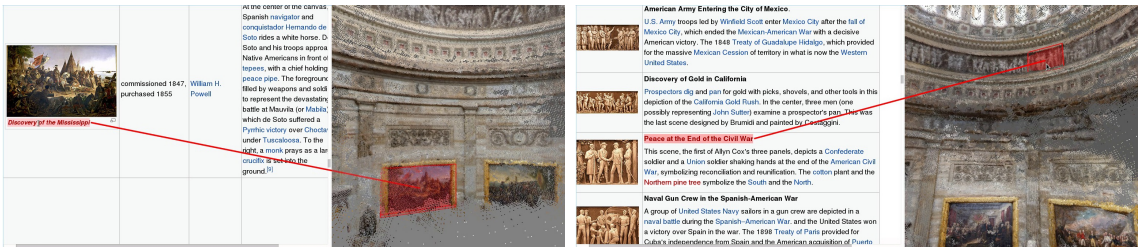
(b) Sistine Chapel, Vatican City



(c) Trevi Fountain, Rome



(d) Pergamon Altar, Berlin



(e) US Capitol Rotunda, Washington D.C.

Figure 4.5: Screenshots of our visualizations for five different sites. Website and photographs in (c) courtesy of www.aviewoncities.com.

by providing the spatial context that relates them.

4.4.2 3D-to-text navigation

You can also navigate in the 3D visualization; dragging the mouse or using the mouse wheel changes the camera viewpoint. When the mouse pointer moves over the projection of the 3D bounding box of a discovered object, the visualization fades in the object's bounding box, hinting that you have found an object. After a short delay, the visualization automatically scrolls the website to show the corresponding object tag in the text pane, highlights it, and draws a connecting line between the tag and the object's bounding box across the two panes. In this way, you can learn about the objects in the scene by simply clicking on the areas of the 3D model and reading the text surrounding the highlighted object tag.

4.4.3 Tour navigation

In most authoritative text sources objects are not described in a random fashion, but follow a sensible pattern around the site. For example, the Wikipedia article of the Pergamon Altar describes the different sculptures in the Gygantomachy frieze from left to right. The Pantheon Wikipedia article first describes the apse, then the chapels and niches on the right side of the apse, followed by the ones on the left side. We can exploit this text structure to create more seamless navigation experiences, where an automated sequence of transitions between relevant images is played as the user reads through the text.

When the user activates the tour mode, a thin highlighting box appears over the text that covers the width of the pane. As the user scrolls through the text, object tags that enter the highlighting box cause the visualization to highlight the tags and automatically move the camera to show a close-up picture of the highlighted object. In this way, the camera automatically follows the exposition.

4.4.4 Implementation details

We used publicly available bundle adjustment and multi-view stereo software to automatically create the 3D models from Internet photo collections using VisualSFM [116, 117, 118] for generating a sparse point cloud, followed by PMVS [35] for generating a dense point cloud. As a post-processing step to filter noisy 3D points from the PMVS output, we apply Poisson Surface Reconstruction [58] to generate a mesh. We then delete small connected components of the mesh and vertices that lie far away from the PMVS points. Finally, we color the mesh vertices according to the closest PMVS points and keep the vertices of the mesh as our final point cloud. Although we generate colored meshes, we only use the vertices from the mesh for visualizations as we found the point cloud is visually more forgiving of artifacts; it avoids the uncanny valley effect and looks better than the mesh.

To highlight the objects in the 3D model, we generate 3D bounding boxes for each object that are rendered semi-transparently in our visualizations. First, we automatically estimate the principal axes of the 3D reconstructions using the recovered 3D scene information. We seek to estimate 3D bounding boxes that maximally align to the dominant orientation of the object, while remaining aligned to the principal axes. We first compute the mode $\mathbf{m}$ over the distribution of the normals of the points that lie in the frustra of the images. We then choose a coordinate unit-vector $\mathbf{x}$ in the world-coordinate frame of the reconstruction that is approximately orthogonal to the mode, preferring the z-axis over the x or y-axes. Finally, we calculate the other axis vector $\mathbf{y} = \frac{\bar{\mathbf{y}}}{\|\bar{\mathbf{y}}\|}$ with $\bar{\mathbf{y}} = \mathbf{m} - (\mathbf{m} \cdot \mathbf{x})\mathbf{x}$ and $\mathbf{z} = \mathbf{x} \times \mathbf{y}$. This approach produces compelling bounding boxes as seen in Figure 4.5.

4.5 Evaluation

As the different navigation modes for the visualization tool depend on the quality of the automatically generated text-object correspondences, we manually evaluate the accuracy of the correspondences. To measure performance, we collected reference texts and computed 3D models for 5 sites, which are listed in Table 4.1.

Site	Pantheon, Rome	Trevi Fountain	Sistine Chapel	US Capitol Rotunda	Pergamon Altar
# 3D points	146K	208K	121K	84K	55K
# ground truth	31	16	31	38	49
# noun phrases	1796	821	3288	2179	2949
# image matches	510	348	2282	884	1600

Table 4.1: Site statistics: # 3D points – number of points in 3D model, # ground truth – number of labeled ground truth objects, # noun phrases – number of automatically extracted noun phrases using the Stanford parser [61], # image matches – number of noun phrases with an image returned from Google image search that aligned to the 3D model. When compared to the number of labeled ground truth objects, there are a large number of (spurious) automatically generated candidate detections (# image matches) that we must cope with.

For Trevi Fountain, we used three webpages[2]. For the remaining sites, we extracted text from their corresponding Wikipedia pages[3]. We evaluate performance relative to a set of *canonical views*, which are a set of registered panoramas or perspective photographs depicting most or all of a site. To define the gold standard, we manually labeled the name and a bounding box for all notable and well-described objects in the reference text using LabelMe [92].

We use a modified Pascal VOC criteria [31] to score detections. First, we relax the relative overlap score to a lower value and require that the center of the detection window lies inside the ground truth. This is to account for the bias in how photographers frame the object in an image (typically not exactly cropped to the object). Moreover, we find that in our visualization task the relaxed bounding boxes still provide useful information about the objects in the scene. Second, we require that at least one of the returned words match the ground truth object tag after stop word removal. This is to handle the noisy nature of object tags where there can be many equally good descriptions, such as "a statue of the Madonna of the Rock by Lorenzetto", "Madonna of the Rock", "Modonna of the Rock by Lorenzetto".

We report site cross validation numbers, where the linear classifier is tested on a single site after being trained on all of the others. We return the top 37% scoring detections after non-maximum suppression. We found that this threshold provides a good balance between recall (many good detections) and precision (good accuracy).

We show example system outputs for the top most confident detections on the canonical views in Figures 4.6-4.8. Correct detections are shown in green and false positives in red. We also display the returned object tags near the corresponding detection window.

In scoring the detections, we found that synonyms in the object tags posed a problem. For example, the central statue in Trevi Fountain can be referred to as "Neptune" or "Ocean". We included such detected synonyms in the ground truth object tags. For scoring detections,

[2]http://www.aviewoncities.com/rome/trevi.htm,
http://www.trevifountain.net, http://www.rome.info/sights/trevi-fountain

[3]We used the following Wikipedia articles: Pantheon, Rome; United States Capitol rotunda; Pergamon Altar; Sistine Chapel ceiling.

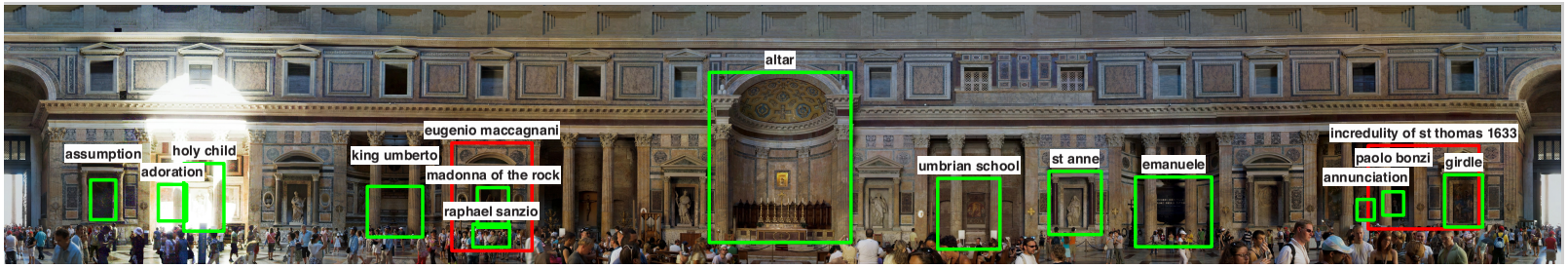


Figure 4.6: Output detections for Pantheon, Rome using named objects automatically extracted from the reference text. Green – correct detection, with the returned object label overlaid. Red – false positives. Photograph by Patrick Landy.

Figure 4.7: Output detections for Sistine Chapel. Photograph by Roy Zipstein, 360Cities.net.

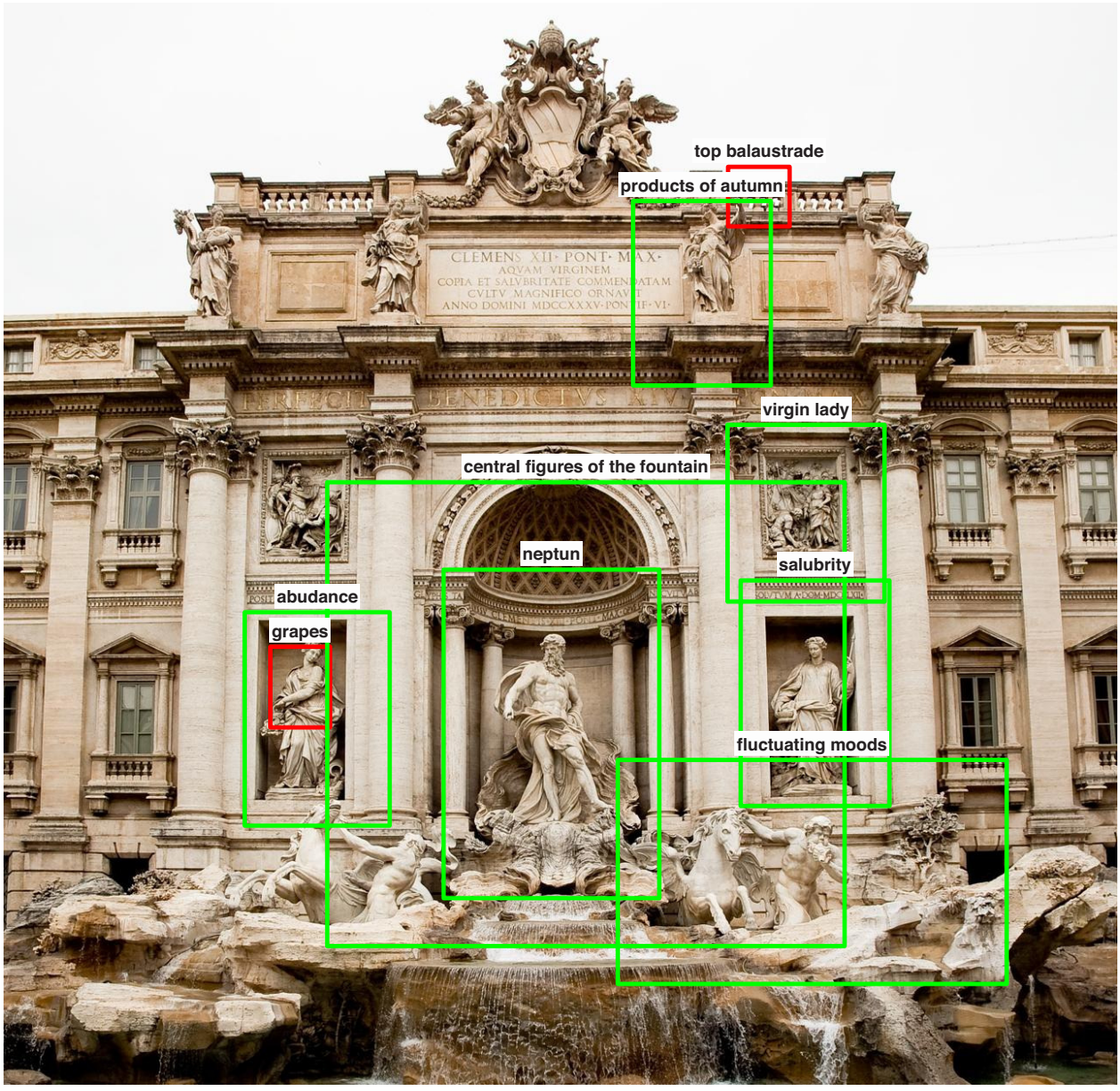


Figure 4.8: Output detections for Trevi Fountain. Photograph by garygraphy.

we ignore any additional detections due to a synonym, after the most confident detection. In this way, a synonym detection does not help or hurt detection performance.

Table 4.2 reports detection precision and recall over the returned detections for all sites. Most recall numbers range from 0.18 for Pergamon Altar to 0.39 for the Pantheon. The Sistine Chapel is a notable exception, which has 0.71 recall. This is mostly due to the fact that the site is well catalogued and features many close-up images available on the Internet. While these recall numbers may seem low, they are quite reasonable in the context of our visualization application. Note, for example, the current Pantheon Wikipedia page includes captioned close-up photos of only **one** out of 31 objects–a recall of only 3%. Our automatic procedure boosts this to 39% and provides a better integrated 3D experience, i.e., in the context of our application, it is not critical to detect *every* object in the scene to provide a compelling and useful visualization experience. More important is that we capture the most important objects in the scene, and important objects tend to be well cataloged with labeled photos on the Internet (which lead to better recall in our system).

For detection precision, we report two numbers: the *raw precision*, which is the proportion of correctly localized objects using the manually labeled ground truth as a guide; and the *full precision*, which uses manually verified detections. The latter is motivated by the fact that often the returned detections correspond to smaller parts of the scene, such as the *trumpets* in *Michelangelo's Last Judgement*, link to relevant descriptive text for other objects, such as *Garden of Eden* for *The Temptation and Expulsion*, or refer to a generic object category, such as *wall frescoes* in the Sistine Chapel. Including these additional detections, we achieve a full precision ranging from 0.65 for US Capitol Rotunda to 0.94 for Pergamon Altar. We believe that this accuracy is reasonable for our 3D visualization, as it allows the user to browse the text and scene with a minimum of incorrect correspondences.

To put these numbers in context, as a baseline, we also computed object recall using the tags associated with the Flickr images that were used to build the 3D models. This Flickr baseline is an upper bound on prior work that segments and labels 3D point clouds by analyzing SIFT feature co-occurence in tagged Flickr photos [101]. We computed the

Site	Pantheon, Rome	Trevi Fountain	Sistine Chapel	US Capitol Rotunda	Pergamon Altar
Recall	0.39	0.31	0.71	0.21	0.18
Raw Precision	0.80	0.31	0.46	0.35	0.56
Full Precision	**0.87**	**0.78**	**0.79**	**0.65**	**0.94**

Table 4.2: Detection accuracy. We measure the proportion of detected objects that are correctly localized in the scene (precision – 1.0 is optimal) and proportion of ground truth objects in the scene that are detected (recall – 1.0 is optimal). Chance is negligible, being proportional to the number of words or phrases on the input text. We report two precision numbers: the *raw precision*, which is the proportion of correctly localized objects using the manually labeled ground truth as a guide; and *full precision*, which uses manually verified detections corresponding to smaller unlabeled parts of the scene, such as the trumpets in *Michelango's Last Judgement* (see text).

proportion of ground truth objects that find a Flickr image whose tag overlaps at least partially with the ground truth tag and depict the ground truth object. We report the object recall for the sites in which we retained the Flickr tags: Pantheon – 0.06; Trevi Fountain – 0; US Capitol Rotunda – 0.21. Notice that the Flickr baseline performs significantly worse for the Pantheon and Trevi Fountain. On the other hand, the US Capitol Rotunda appears to be well-documented on Flickr and achieves the same recall as our full system, with many of these images appearing in the query expansion step. However, it is not straightforward to filter the many false positive tags that appear in the Flickr image set.

4.5.1 Error Analysis

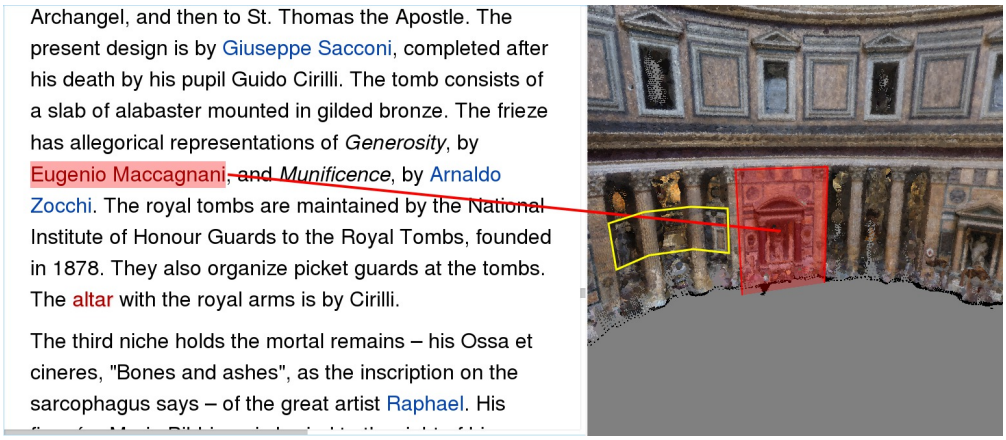
We have observed different sources of errors of our system, which result in inaccurate labels returned by our system and missed objects. We describe these errors and illustrate them in Figure 4.9.

One common case is when text spans are paired with bounding boxes that contain the named object, but are too large. This happens when Google image search returns images that correctly depict the object of interest, but are not tightly cropped to the object. For example, in Figure 4.9(a) the bounding box for the painting *The Incredulity of St. Thomas* is large and encloses the object itself, along with the first niche and the first chapel.

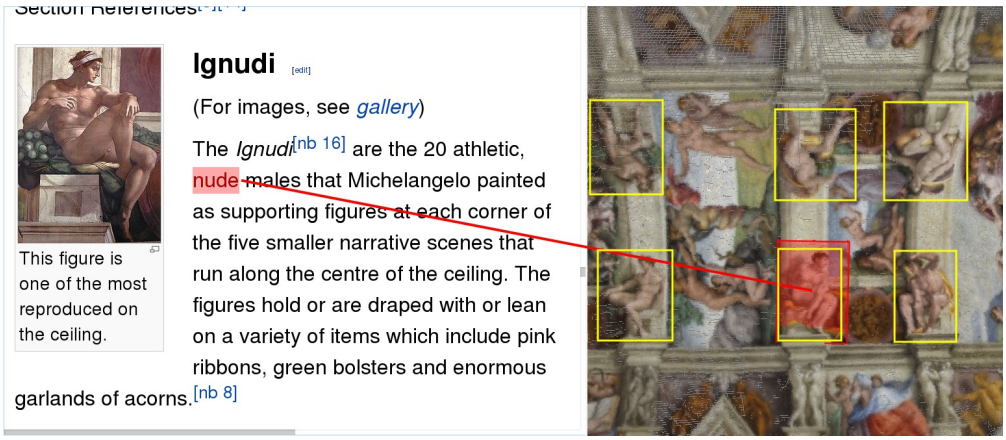
We have also observed incorrect object correspondences, such as the one shown in Figure 4.9(b). The recovered bounding box for the object *Eugenio Maccagnani* encloses the niche around the *tomb of Raphael*, which is described in the following paragraph in the text. These errors typically come from noisy co-occurrences between images and text in the online sources.

A challenging case is when an object is not a specific instance, but belongs to a set, as shown in Figure 4.9(c). Here, the *ignudi* describes the set of depicted nudes in the Sistine chapel. Our system cannot identify all of them since the system assumes a one-to-one correpondence between a named object and its depiction in the 3D scene. While we could relax this constraint, it would result in lower overall precision due to the noisy results of the

(a) Bounding box is too large



(b) Incorrect object tag



(c) Multiple object class instances

Figure 4.9: Failure cases: obtained bounding box is shown in red and correct objects are shown in yellow.

Google image search.

In addition, we have observed failures that result in object misses. These are primarily due to: (1) incorrect images that are returned from Google image search for a candidate object, and (2) when the object is poorly reconstructed during the structure-from-motion step, causing the Google images not to match. This can be partially remedied by better online documentation of the objects and improved 3D models.

We find that there is evidence in the text to cope with some of these errors. For example, the *Incredulity of St. Thomas* is described to be "on the right wall" of the *Chapel of the Annunciation*; there is a clear description of the *ignudi* being multiple figures: "the Ignudi are the 20 athletic, nude males." Also, there is often information in the text about the class of the objects, e.g. a named object can be described as being a *painting* or *statue*. The category of the object could be extracted from the text and used with object detectors trained for the category. Moreover, bottom-up segmentation could be used to improve object localization. Developing a model that could incorporate such cues is an important area for future work.

4.6 Conclusion

This chapter introduced the first system capable of using online text and photo collections to automatically build immersive 3D visualizations of popular sites. These included a number of new interactions: text can be selected to move the camera to the corresponding objects, 3D bounding boxes provide anchors back to the text describing them, and the overall narrative of the text provides a temporal guide for automatically flying through the scene to visualize the world as you read about it.

While our system is built using off-the-shelf ingredients, we argue that the ideas and the system are new. In particular, we introduce (1) the concept for a 3D Wikipedia based on crowd-sourced data on the Internet, (2) the insight of using text parsing + Google image search to connect web text to 3D shape data, and (3) a viable approach that accomplishes this goal, incorporating a series of sophisticated steps (3D reconstruction, text parsing, and a classifier to improve precision). Experiments on multiple sites demonstrate that this approach

has consistently high precision, which is crucial for enabling high quality interactions, even if all objects are not yet recognized. Our current system works on the most popular sites, as it requires lots of images and good text. Going forward, with growth in photo and text corpi, the system will work "as is" for more scenes as the underlying data improves. Improvements to 3D reconstruction algorithms and text parsers will also further improve applicability.

While the results are encouraging, there is room for improvement. While improvements in search technology will reduce false negatives (missed objects), we have barely tapped into the structure and constraints expressed in the text, which have significant potential to reduce false positives (mislabeled objects). For example, one especially promising topic for future work will be to leverage *spatial* terms (e.g., "in the first niche to the left of the door...", "the painting above the altar...") to constrain the placement of objects in the scene. Developing semi-automated methods that leverage people to assist in the labeling task is another interesting topic of future work.

Chapter 5

THE 3D JIGSAW PUZZLE

Recent breakthroughs in computer vision now allow us to model our world in 3D with extraordinary accuracy and visual fidelity from just about any set of overlapping photos [5, 3, 97]. However, a limitation of state-of-the-art 3D reconstruction techniques from Internet photos is that large scenes tend to break up into a collection of disconnected pieces due to gaps in the depicted scene coverage or matching failures. Rather than a single, fully-connected Vatican model, for instance, we get a collection of smaller 3D pieces for different rooms, such as the Sistine Chapel, the Raphael Rooms, and the Hall of Maps, each having their own 3D coordinate system. A major challenge is to automatically put these 3D pieces together correctly into a global coordinate frame. This is akin to solving a *3D jigsaw puzzle*, where the scale, rotation, and translation of the 3D pieces must be recovered with respect to the global coordinate frame.

Solving the 3D jigsaw puzzle is extremely difficult using image information alone due to the aforementioned coverage and matching failures. Instead, we seek to leverage readily available map data to solve the 3D jigsaw puzzle. Such data provides additional information that helps constrain the spatial layout of the 3D pieces. For example, a map of the Vatican shows an annotated floorplan of the different rooms, with a legend providing the names of the rooms and any objects located inside the rooms. Such maps are plentiful and widely available, for example in tourist guidebooks (e.g. Rick Steves, Lonely Planet, Baedeker) and online (e.g. planetware.com).

Automatically leveraging map data for the 3D jigsaw puzzle is challenging as the pieces

---

This chapter describes work that was originally published in the 2014 European Conference on Computer Vision [76].

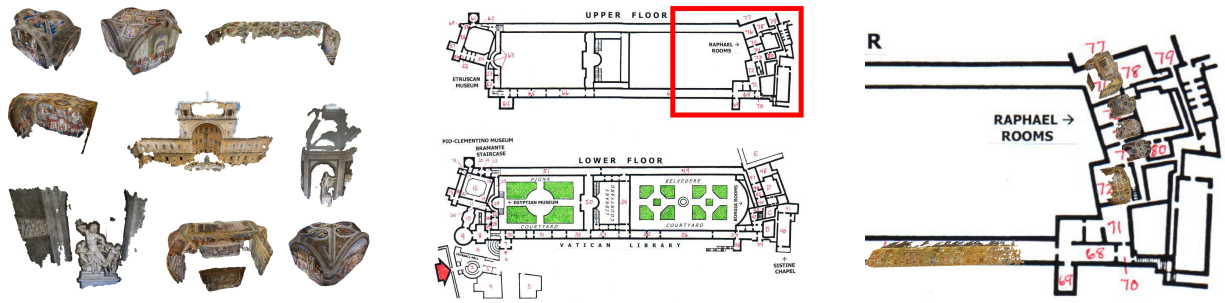


Figure 5.1: Given a set of disconnected reconstructed 3D models of a large indoor scene, for example the Vatican (left), we jointly reason about a map of the site (middle) and the 3D pieces to produce a globally consistent reconstruction of the entire space (blow up at right).

are unlabeled and lack absolute position, orientation, and scale. The 3D Wikipedia system, presented in the previous chapter, provided one approach to automatically link objects described in text to their spatial location in a 3D model [91]. While that approach can be used to link the 3D pieces to text on an annotated map, it does not provide information on how to *place* the pieces in a global coordinate system. Moreover, most maps provide only 2D cues (e.g., via a floor plan), with objects and dimensions placed only approximately. Finally, we must cope with rooms having orientation ambiguities (e.g. square or rectangular rooms), which the map and 3D piece geometry alone cannot disambiguate.

The key to our approach is to extract and integrate position, orientation, and scale cues from the 3D pieces and the map. These include the room shape, map annotations, cardinal direction (available as compass measurements provided in the image EXIF data used to reconstruct the 3D pieces), and crowd flow through the different rooms of the site. The latter crowd flow cue, which measures the dominant direction of travel through the 3D pieces, provides information on the orientation of the pieces. For example, in the Vatican Museum tourists tend to go from the entrance toward the Sistine Chapel, passing through the Gallery of Candelabra, Gallery of Tapestries, and Hall of Maps along the way. We formulate the 3D jigsaw puzzle problem as an integer quadratic program with linear constraints to

globally solve for the 3D layout of the pieces. Ours is the first system to reconstruct large indoor spaces of famous tourist sites from Internet photos via joint reasoning with map data and disconnected 3D pieces returned by Structure-from-Motion.

We show compelling results on four major sites. Our system reliably assigns and orients many of the 3D pieces relative to the input maps (we provide a detailed analysis of assignment precision/recall and orientation accuracy). Moreover, we show an integrated visualization of the map and reconstructed 3D geometry, where a user can interactively browse and fly to different rooms of the site.

5.1 Related work

Our work is related to prior research leveraging auxiliary information, such as geographic data, human path of travel, and text, to augment 3D reconstruction and image localization. Geographic data, such as Google Street View and Google Earth 3D models, has been used to georegister point clouds generated from Internet images [113]. Maps have been used in conjunction with visual odometry for self localization [17, 71]. Human path of travel has been used for image geolocalization [54] and predicting tourists' path of travel [100]. Note that in this work we use human path of travel to recover 3D piece orientation. Finally, text has been used to automatically label objects in reconstructed geometry [91].

Most related is prior work that matched free space from a 3D reconstruction to white space in a map [55]. However, [55] addressed a particularly simple case where the 3D jigsaw puzzle has only one large piece (the 3D model), and the floor plan is accurate. While aligning 3D geometry shards has been explored by other authors (e.g., Stanford's Forma Urbis project [63]), our problem is more challenging as the scale of each piece is unknown and we do not have access to complete scans. Also related are approaches for solving 2D jigsaw puzzles [21, 22, 36], which operate entirely on 2D rectangular puzzle pieces and determine puzzle piece locations through absolute position cues (e.g. corners, edges, color) and adjacency cues (e.g. shape). The analogy in our case is that label correspondences provide absolute cues and tourist flow provides adjacency.

Reconstructing large indoor spaces is a challenging problem due to lack of texture on many surfaces and the difficulty of systematically scanning every surface of a site. Major efforts to scan and reconstruct large indoor scenes include the Google art project [40], museum reconstruction via constructive solid geometry [119], and human-operated systems to scan a large site [19, 120].

5.2 System overview

In this chapter, we present a system to solve the 3D jigsaw puzzle via joint reasoning over 3D geometry and annotated map data. Our system takes as inputs: (i) one or more reconstructed 3D pieces for a site and (ii) an annotated map corresponding to a set of 2D map points of interest (associated with rooms and named objects), with corresponding 2D map regions (the extent of rooms and objects in the map) and text annotations (the legend). Section 5.4.2 describes a semi-automatic method to parse an annotated map.

Our system first generates a discrete set of candidate placements of the 3D pieces to the map points of interest. 3D pieces are assigned to the map by querying Google Image Search using the extracted text annotations from the map and linking the returned images to the 3D pieces via camera resectioning. This provides links between a given map point of interest to candidate 3D locations on the 3D pieces. Note that the links are noisy as the returned images may depict incorrect locations of the site. Given the links, a discrete set of candidate 3D transformations to the global coordinate system are generated for each 3D piece.

Given the candidate placements, we optimize an objective function that seeks a globally consistent layout of the 3D pieces by integrating cues extracted over the points of interest, their 2D map regions, and the 3D pieces, described in Section 5.4. The objective integrates cues about the shape of the rooms, cardinal direction, crowd flow through the site, and mutual exclusion of the 3D pieces. We show results of our system in Section 5.5.

5.3 Map parsing

Given an annotated map of a site, we seek to extract the spatial layout of the different rooms and objects depicted on the map. Automatically parsing a map is an interesting problem, but not strictly necessary for our task, as it would be straightforward to have manual workers parse maps for all leading tourist sites, or have future map-makers generate maps with the requisite annotations. Here, we describe a semi-automatic method of extracting the spatial layout and the object labels. We have restricted ourselves to annotated maps depicting the floor plan of a space, with referenced rooms and objects in the map appearing as text in a legend, as illustrated in Figure 5.6.

Our map parsing procedure begins by recovering a set of 2D regions from the floor plan corresponding to rooms, hallways, courtyards and other features of the site. We extract the floor plan of the map by clustering the pixel values found in the map image by K-means. We generate 2-4 clusters and manually select the cluster corresponding to the floor plan to form a binary image. To extract regions corresponding to the rooms we must close small gaps in the floor plan corresponding to doors and passages, which we achieve by simple morphological operations We recover a segment for the room region by flood filling seeded by the room annotation marker on the map.

While OCR systems (e.g. Tesseract [111]) have shown much success in reading text in images, automatically recognizing text labels and markers in maps is still very difficult since the text is not generally structured into lines and may appear in different orientations, thus violating critical assumptions made by these systems. Moreover, markers and other visual elements appearing on the floor plan confuse the text line detection algorithms. The application of recently developed scene text recognition systems [15, 30, 39] to annotated maps remains outside the scope of this work and an interesting topic for future work. For our purposes we have manually annotated the map using LabelMe [92] by marking each text label or marker with the appropriate text label.

5.4 Model for the 3D jigsaw puzzle

Given a discrete set of candidate placements of 3D pieces to map points of interest, we seek a globally consistent layout of the 3D pieces. Let $p \in \{1, \ldots, P\}$ index the map points of interest, $m \in \{1, \ldots, M\}$ the 3D models, $q_m \in \{1, \ldots, Q_m\}$ 3D locations on 3D model m, and $t_m \in \{1, \ldots, T_m\}$ candidate 3D transformations of 3D model m to the global coordinate system. A candidate placement is the tuple (p, m, q, t), where we omit the subindices for brevity.

A solution to the 3D jigsaw puzzle is a selection of 3D piece placements from the candidate set. We define binary variables $x_{p,m,q,t} \in \{0, 1\}$ to indicate whether the candidate placement appears in the solution set and auxiliary binary variables $y_{m,t} \in \{0, 1\}$ to indicate that 3D model m is placed in the global coordinate system under 3D transformation t. We formulate the 3D jigsaw puzzle as an integer quadratic program with linear constraints where vector b and matrix A encode unary and pairwise cues over the position, scale, and orientation of the candidate placements (described in Section 5.4.1):

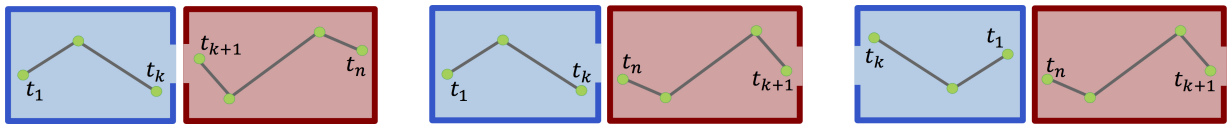
$$\max_{x,y} \quad x^T A x + b^T x \tag{5.1}$$

$$\text{s.t.} \qquad \forall p \quad \sum_{m,q,t} x_{p,m,q,t} \leq 1 \qquad\qquad \forall q \quad \sum_{p,m,t} x_{p,m,q,t} \leq 1 \tag{5.2}$$

$$\forall m \quad \sum_{t} y_{m,t} \leq 1 \qquad\qquad \forall p, m, q, t \quad x_{p,m,q,t} \leq y_{m,t} \tag{5.3}$$

Constraints (5.2) enforce mutual exclusion of the 3D puzzle pieces. We require each point of interest p to be assigned to at most one 3D location q on a model, and vice versa. We find that enforcing mutual exclusion is critical for our problem since we are reconstructing unique object instances of a site. Constraints (5.3) enforce each model m to be placed in the global coordinate system under a single 3D transformation t.

Given pairwise and unary coefficients A and b, we optimize Objective (5.1) using mixed-integer quadratic programming [11]. Note that while it has been shown that solving jigsaw puzzles with uncertainty in the piece compatibility is NP-hard [27], the small size of our



(a) (b) (c)

Figure 5.2: Illustration of the crowd flow cue for two adjacent rooms on the map. For a sequence of photos captured by a particular user, green points show the location where the images where taken and $t_1, \ldots, t_n$ their ordered time stamps. Here, the user moved from the blue to the red room from left to right. Our goal is to orient the rooms to be consistent with the direction of travel. (a) room orientations are consistent with the user path through both rooms. (b) the red room on the right is inconsistent with the user path. (c) both rooms are inconsistent with the user path.

datasets, of up to a few dozen pieces, enables us to express the mutual exclusion constraints exactly. This is in contrast to recent work in modeling 2D jigsaw puzzles that have formulated the problem as a Markov Random Field with mutual exclusion constraints approximated by a set of local pairwise terms due to large problem size [21, 22].

5.4.1 Cues for position, scale, and orientation

In this section we describe the cues that are used to pose the 3D pieces relative to the map. These cues encode the crowd flow through the space, number of registered image search results to 3D pieces, cardinal direction of the pieces, and room shape.

Crowd flow potential: As previously noted [100], for many popular places people tend to visit different parts of the scene in a consistent order. For example, in the Vatican Museum, most tourists walk from the entrance toward the Sistine Chapel, passing through the Gallery of Candelabra, Gallery of Tapestries, and Hall of Maps along the way. We seek to harness the "flow of the crowd" to help disambiguate the orientation of the 3D pieces.

We wish to characterize the crowd flow within each 3D piece m and between 3D pieces m and m'. We start by considering the sets of photos taken by individual Flickr users that

were aligned to the 3D pieces and sort the photos based on their timestamps. These aligned images indicate the users' direction of travel within the 3D pieces (e.g., tourists move from right to left of the main painting inside the Hall of the Immaculate Conception) and across the 3D pieces (e.g., tourists visit the Galeria of Candelabra before the Gallery of Maps). We say that the candidate placements of two 3D pieces agree with the crowd flow if the dominant direction of travel *across* the two pieces is oriented in the same direction as *within* the pieces after placing them onto the global coordinate system. We illustrate the crowd flow cue in Figure 5.2.

More concretely, given the camera locations for the images for a particular user i in model m, let $d^m_{i,k}$ be a unit vector in the direction of travel between consecutive images k and $k+1$ in the sequence, which corresponds to how the user moved between shots. For candidate placement $\alpha = (p, m, q, t)$, we define the dominant direction of travel within 3D piece m as $\delta_\alpha = H_t(norm(\sum_{i,k} d^m_{i,k}))$ where $H_t(\cdot)$ is the 3D transformation for t and $norm(\cdot)$ normalizes the input vector to unit length.

To estimate the dominant direction of travel across 3D pieces m and m', we count the number of users $u_{m,m'}$ that took a picture first in m and later in m'. For candidate placements α and α' with $m \neq m'$, we denote the dominant direction of travel across the two pieces in the global coordinate system as the unit vector $\delta_{\alpha,\alpha'} = sign(u_{m,m'} - u_{m',m}) \cdot norm(H_{t'}(c_{m'}) - H_t(c_m))$ where c_m is the 3D centroid of 3D piece m. Note that if most users travel from m to m', $\delta_{\alpha,\alpha'}$ will point in the direction from 3D piece m to m' in the global coordinate system. We define the crowd flow cue for candidate placements α and α' as the sum of inner products:

$$A_{\alpha,\alpha'} = < \delta_{\alpha,\alpha'}, \delta_\alpha > + < \delta_{\alpha,\alpha'}, \delta_{\alpha'} > \tag{5.4}$$

Unary potentials: For each candidate placement we extract unary potentials for assignment $\phi^{assign}(\alpha)$, cardinal direction $\phi^{card}(\alpha)$, and room shape $\phi^{shape}(\alpha)$. We concatenate these potentials into vector $\Phi(\alpha)$ and, given weights w, define the unary coefficients b as:

$$b_\alpha = w^T \Phi(\alpha) \tag{5.5}$$

We wish to leverage the vast amounts of labeled imagery online to connect the map points of interest to their locations in the 3D pieces. Using the text annotation for each point of interest in the map, we issue a query to Google Image Search concatenating the annotation text with the site name, followed by registering the returned images to the 3D pieces. We define $\phi^{assign}(\alpha) = count(p, m, q)$ as the number of images retrieved by querying for the text associated with map point of interest p that are registered to the 3D location q in model m.

A small fraction of Flickr images contain heading information in EXIF tags (e.g., via compass). Although we have found such data to be sparse and not always reliable, we can exploit it when available. The cardinal direction potential $\phi^{card}(\alpha)$ measures the compatibility of compass measurements corresponding to images used to reconstruct a 3D piece to a cardinal direction given on the map (e.g. "north"). Let $C_m > 0$ be the number of images used to reconstruct 3D piece m having a heading and $C_{m,t}$ be the number of such images that agree on the orientation of the provided cardinal direction within τ degrees after applying 3D transformation t into the global coordinate system. We define the potential to be $\phi^{card}(\alpha) = C_{m,t}/C_m$.

Next we wish to encode how well the 3D piece matches the shape of a given 2D region on the map. We encode the shape by projecting the Structure-from-Motion points of model m onto the map via transformation t and rasterize the points into a grid of cells. The shape potential $\phi^{shape}(\alpha)$ is a weighted sum of three terms: (i) the ratio of intersection area over union between the 2D region and occupied grid cells, (ii) average truncated distance of each grid cell to the 2D map region edge, and (iii) fraction of grid cells that lie outside of the region.

5.4.2 Generating candidate placements

In this section we describe how to generate the set of candidate placements of 3D pieces to map points of interest. First, we parse the map into a set of regions and points of interest with accompanying text, as described in Section 5.4.2. Then we describe how we assign and align the 3D pieces to the map regions and points of interest.

Figure 5.3: Left: A 3D piece of our system corresponding to the Hall of the Immaculate Conception. Middle: Colored 2D regions extracted from the floor plan. Number 72 in purple corresponds to the ground truth location of the 3D piece. Right: Candidate placements of the 3D piece to the 2D region.

Given the extracted text annotations from the map, we align images downloaded from Google image search to the 3D pieces. We cluster the set of inlier 3D points across all queries and set the 3D locations q as the centers of mass of the clusters. We orient the vertical direction of each 3D piece by aligning its z-axis with its up vector and setting the ground plane ($z = 0$) at the bottom of the piece. The up vector is the normal direction of a plane fitted to the inlier camera centers of the piece, oriented towards the cameras' up vectors.

A map may provide labels for only the room and/or for multiple objects in a room. For example, the Vatican Museums have only the rooms labeled, whereas the Pantheon has objects labeled within the main room. We wish to account for both cases when generating candidate placements. When only the room is labeled, we generate multiple candidate placements by finding local maxima of the unary shape potential $\phi^{shape}(\alpha)$. When multiple objects are labeled, we use the candidate assignments between the 3D locations on the models and the 2D points of interest on the map as putative matches. We then estimate a similarity transformation given the matches to yield the candidate placements. Example candidate placements are shown in Figure 5.3.

Site	# POIs	# GT POIs	# GT Orientations	# Images	# 3D Pieces
Vatican Museums	75	30	11	11K	68
Hearst Castle	22	5	5	3K	30
Pantheon	9	8	8	705	11
St. Peter's	34	13	11	3K	55

Table 5.1: Site statistics: # POIs – number of points of interest in the map, # GT POIs – number of points of interest in the map with ground truth 3D model assignments, # GT Orientations – number of points of interest in the map with ground truth 3D model orientation assignments, # Images – number of images used in the 3D reconstruction, # 3D Pieces – number of reconstructed 3D pieces.

5.5 Results

We evaluated our system on four major tourist sites: the Vatican Museums, St. Peter's Basilica in Rome, Pantheon in Rome, and the Hearst Castle. We collected maps for each site and reconstructed 3D models for the sites by downloading images from Flickr by querying for the site name and running VisualSFM [118, 117]. In addition, for each reconstructed Flickr photo, we downloaded all photos taken by the same user within 2 hours and match them to the reconstructed pieces, yielding a much larger image set (factor of 5-10). For visualization purposes we use PMVS for multi-view stereo [35] and Poisson Surface Reconstruction [58] to generate colored meshes. Note that all these packages are freely available online.

We collected ground truth assignments between the pieces and the map legends by finding information in authoritative sites, such as Wikipedia articles and specialized sites about the landmarks, like the official website of the Vatican Museums or saintpetersbasilica.org. Collecting ground truth orientations of the 3D pieces is challenging given that images alone do not disambiguate between orientations. Fortunately some authoritative sites contain more detailed maps for a small section of a landmark that place different objects inside the rooms

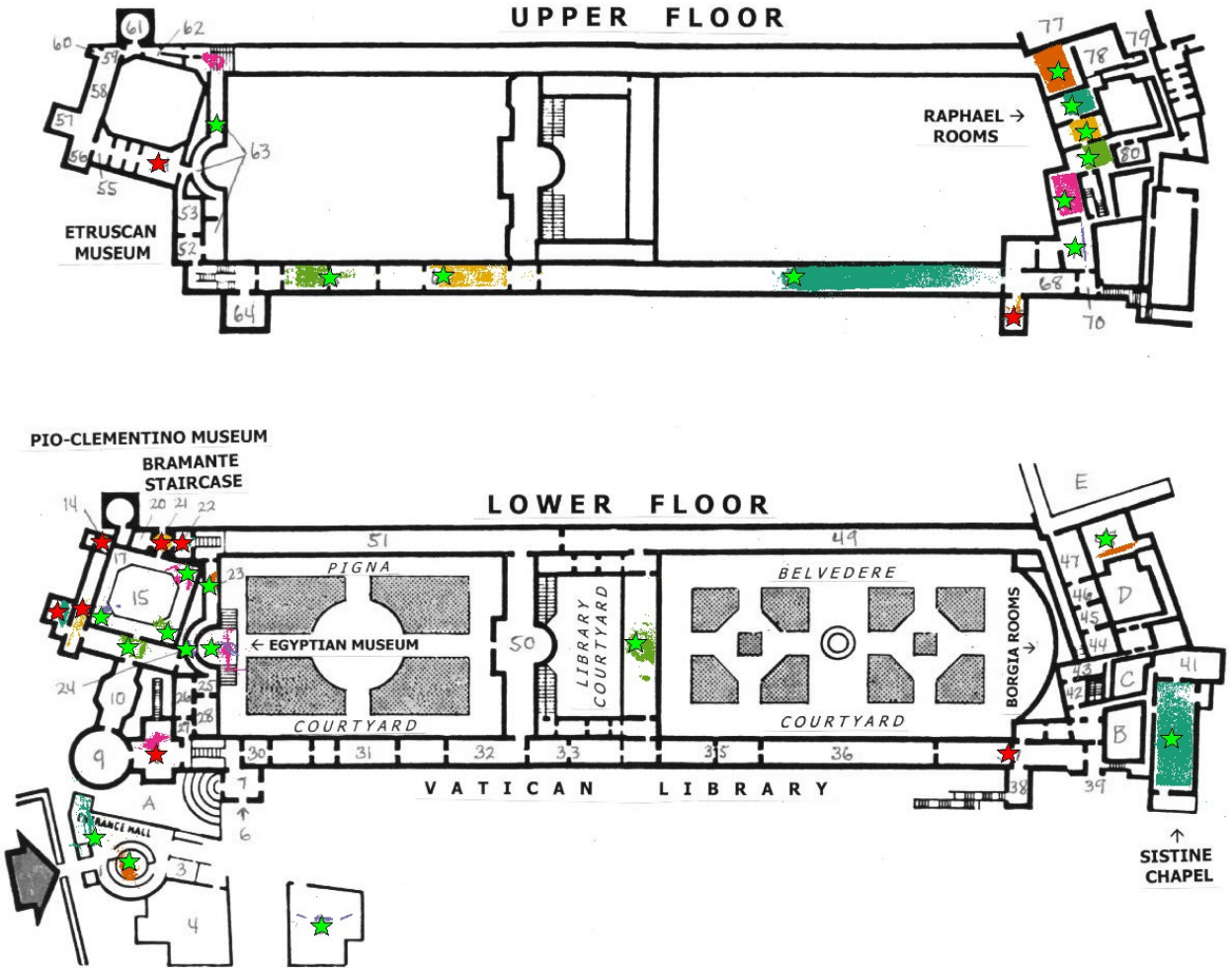


Figure 5.4: Results for the Vatican Museums. 3D pieces are shown as the projection of the SfM points on the map, with different colors for each model. Green stars represent correct assignments, red stars incorrect ones. Please zoom in on the electronic version to see the details.

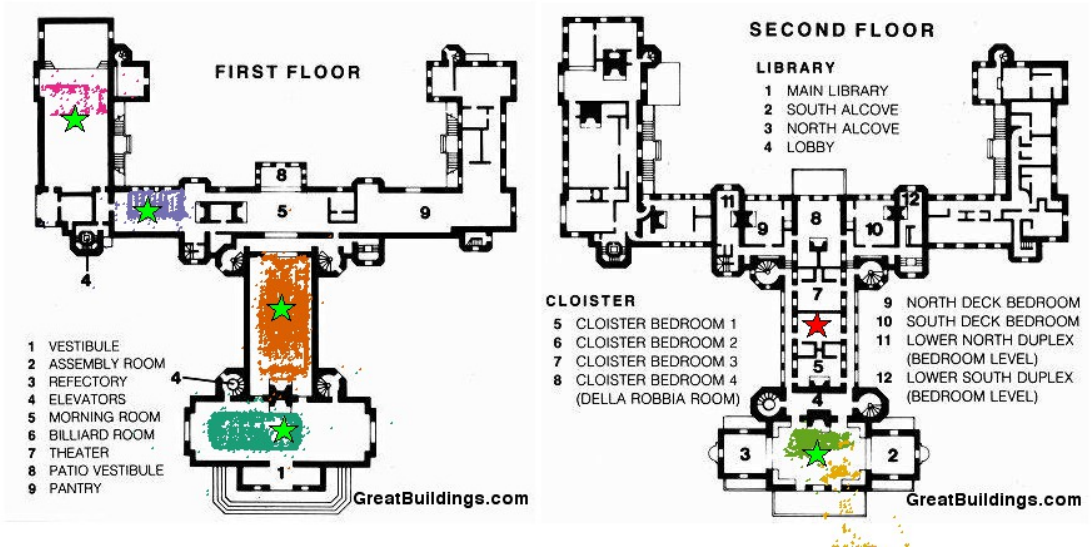


Figure 5.5: Results for the Hearst Castle. 3D pieces are shown as the projection of the SfM points on the map, with different colors for each model. Green stars represent correct assignments, red stars incorrect ones.

or enumerate the views with their cardinal orientations. We can also infer the orientation of some rooms from official museum itineraries by correlating the direction of travel of the 3D pieces with the observed direction of travel from the Flickr users. We summarize the ground truth dataset statistics in Table 5.5.

The Vatican Museums and the Hearst Castle datasets are examples of very large multiroom scenes where most pieces correspond to complete rooms in the site, like the Sistine Chapel or the Raphael Rooms in the Vatican Museums. Figures 5.4 and 5.5 show the recovered layout of the different 3D pieces using the annotated maps for the Vatican Museums and Hearst Castle, respectively. Notice that we are able to correctly position and scale/orient many of the 3D pieces. While our 3D model coverage appears sparse in some regions, particularly the lower floor of the Vatican and 2nd floor of Hearst Castle, we correctly place most of the most visited and well-photographed rooms, such as the Raphael Rooms and the 2nd floor galleries of the Vatican Museums. Indeed, the correctly aligned pieces account

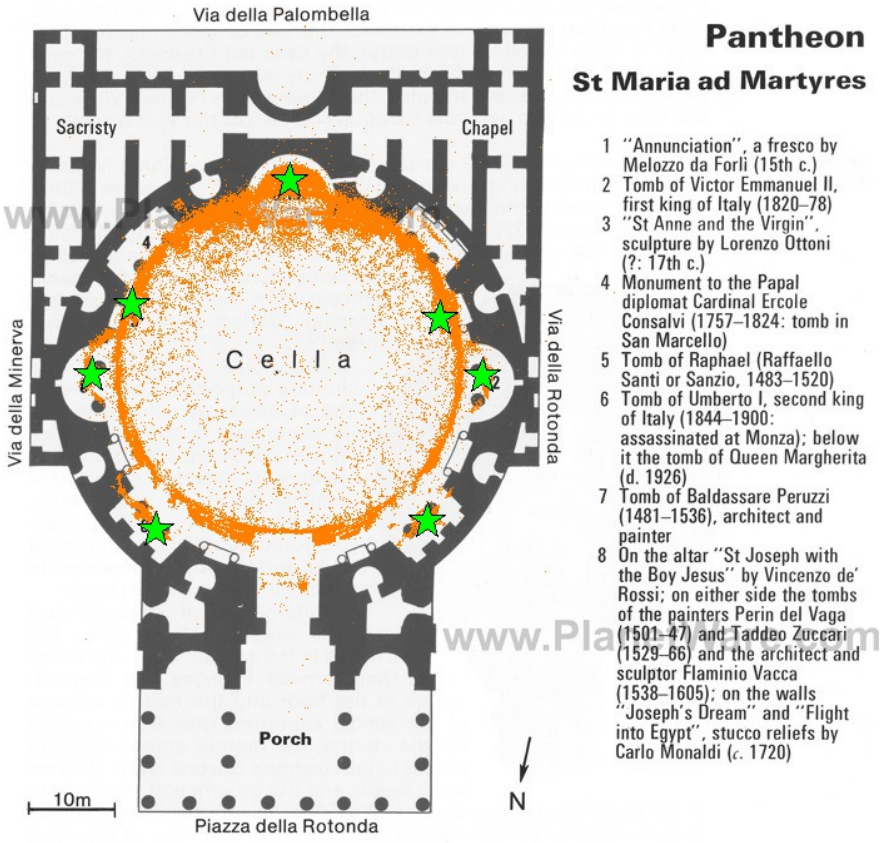


Figure 5.6: Results for the Pantheon. 3D pieces are shown as the projection of the SfM points on the map, with different colors for each model. Green stars represent correct assignments, red stars incorrect ones.

for 75% and 73% of all reconstructed images for the Vatican Museums and Hearst Castle respectively. Note that some pieces are incorrectly scaled, like the Pigna Courtyard, due to the lack of a complete model of the room, as well as errors in the map parsing.

The Pantheon and St. Peter's Basilica are examples of single large rooms, where the annotated maps detail the specific objects names present in the site. Both sites contain large open spaces that enable the 3D reconstruction process to create a mostly complete 3D model of the entire site. Figures 5.6 and 5.7 show the recovered layout for both sites. The Pantheon model was aligned to the map by the assignment of 7 of its objects to points of

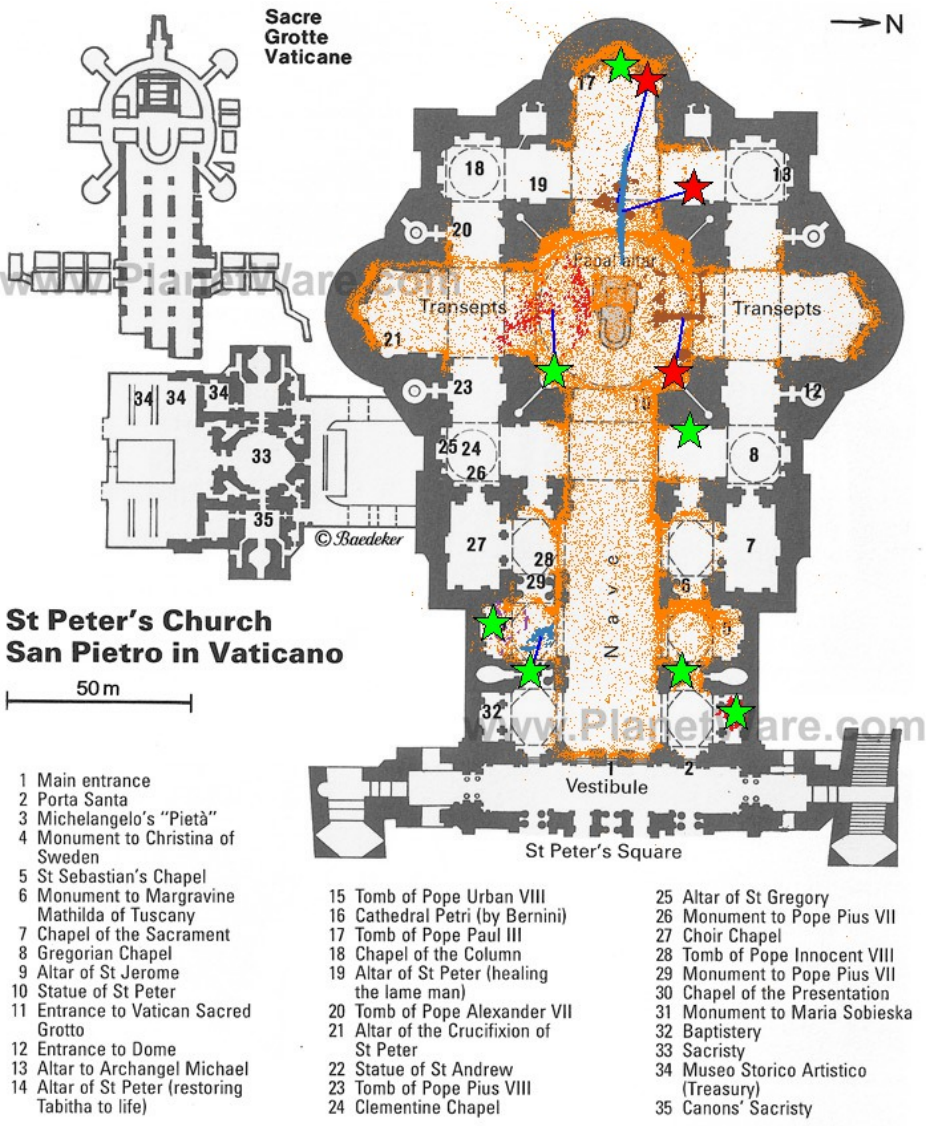


Figure 5.7: Results for the St. Peter's Basilica. 3D pieces are shown as the projection of the SfM points on the map, with different colors for each model. Green stars represent correct assignments, red stars incorrect ones.

Site	Assignment				Orientation
	Baseline		Model		Model
	Precision	Recall	Precision	Recall	Accuracy
Vatican Museums	53%	57%	73%	43%	91%
Hearst Castle	83%	27%	83%	27%	60%
Pantheon	67%	89%	100%	78%	100%
St. Peter's	45%	59%	70%	29%	50%

Table 5.2: For each site, we report assignment precision/recall values with respect to all annotated points of interest in the map for our model and a baseline (see text), and orientation accuracy of our model.

Site	Crowd flow	Cardinal Direction	Joint Model
Vatican Museums	27%	72%	91%
Hearst Castle	40%	40%	60%

Table 5.3: For each site, we report orientation accuracy using the crowd flow cue, the cardinal direction cue and the joint model.

interest in the map. In the St. Peter's case, three objects contained in the large 3D model were assigned to points of interest as well as other smaller models, such as Michelangelo's Pieta and the Chapel of Presentation.

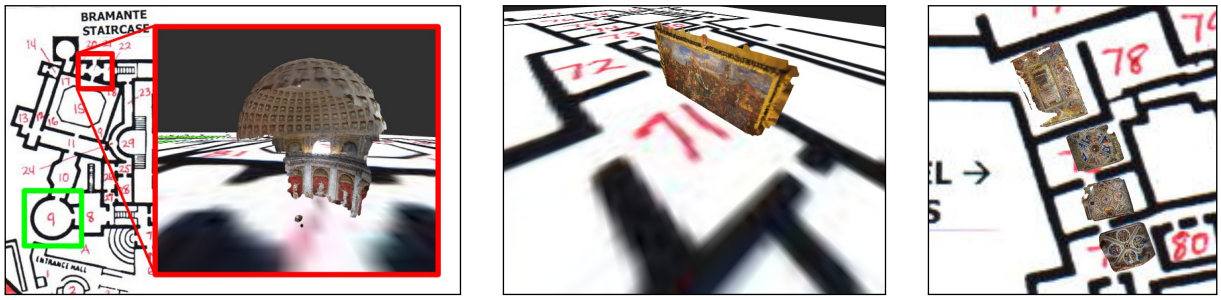
We quantitatively evaluate the assignments of 3D pieces to the points of interest in the map and the orientation of those assignments in Table 5.2. As a baseline we use only the assignment potential score $\phi^{assign}(\alpha)$, which ignores the mutual exclusion constraint. Our system consistently improves the precision of the assignment over the baseline. The orientations proposed by our system for the correctly assigned points of interest are correct in 25 out of 33 cases across all sites.

Site	Users	Photos		Recons. Photos	
		Before	After	Before	After
Vatican Museums	2112	11K	99K	4K	11K
Hearst Castle	367	3K	16K	828	3K

Table 5.4: For each site, we report the number of Flickr users, number of photos before and after the dataset expansion and number of reconstructed photos before and after dataset expansion.

We perform an ablative study over the orientation cues for the sites with multiple rooms (Vatican Museums and Hearst Castle). Note that the Pantheon is a single large room and St. Peter's has stand-alone objects (e.g. the Pieta, the Altar of St. Jerome), plus one central room. In Table 5.3 we show statistics of orientation accuracy values using the crowd flow cue, the cardinal direction cue and the joint model. The crowd flow cue disambiguates cases such as the galleries in the second floor of the Vatican, but fails on 3D pieces representing objects, such as statues or paintings, since users don't move in a predetermined path of travel when photographing them. The compass cue is powerful when enough data is available, but is ineffective for datasets with fewer photos, like the Hearst Castle, where we only match 3 out of the 16 photos with compass heading to the assigned 3D pieces. Augmenting the image dataset by downloading more photos for the set of users is critical for the crowd flow and cardinal direction cues, as it vastly increases the number of reconstructed photos and also the number of reconstructed photos per user. In Table 5.4 we report statistics of the dataset expansion.

For each dataset, the integer quadratic program contained up to a thousand variables and was solved within 5 seconds on a single workstation with 12 cores.



(a) (b) (c)

Figure 5.8: Failure modes: (a) Incorrectly placed 3D model of the "Round Hall"; assigned point of interest marked in red, correct one in green, (b) ambiguous placement of object due to lack of scale and orientation information, (c) inaccurate map with incorrect aspect ratio for the rooms.

5.5.1 Failure cases

We have observed different failure cases of our system, showcasing the challenges of reconstructing indoor spaces from 3D pieces.

In some cases, the annotated text in the map may yield noisy image search results, leading to incorrect assignments. For example, in Figure 5.8(a), we show the model recovered for the point of interest labeled as "Round Vestibule" in the Vatican Museums that is actually the "Circular Hall", which is located in the same Pio Clementino Museum.

Another interesting case are the recovered 3D pieces corresponding to individual objects, such as the painting in the "Sobieski Room", shown in Figure 5.8(b). The room that contains the painting is rectangular and provides no cues for precise alignment of the object, even when the orientation is recovered from heading measurements. Our system can still provide a plausible alignment of the object along one of the walls, but the object might be scaled incorrectly.

Our system also fails to produce precise alignments to the walls of the rooms, such as the "Raphael Rooms" shown in Figure 5.8(c), due to inacurracies of the map. In the annotated

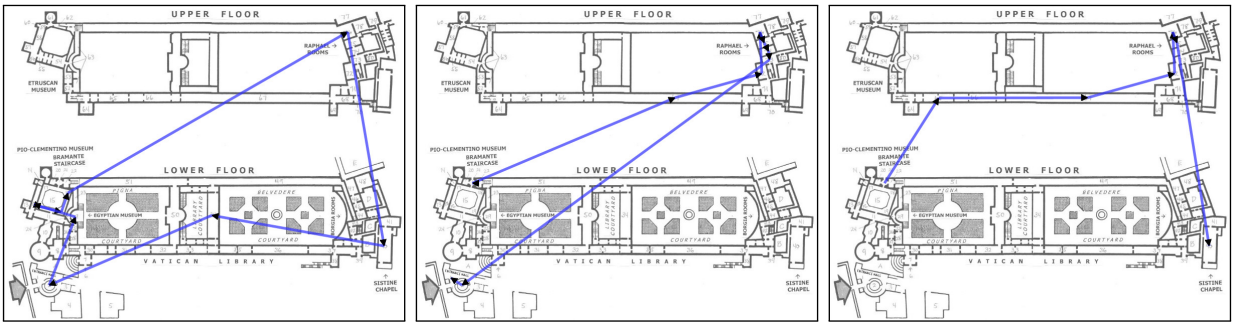
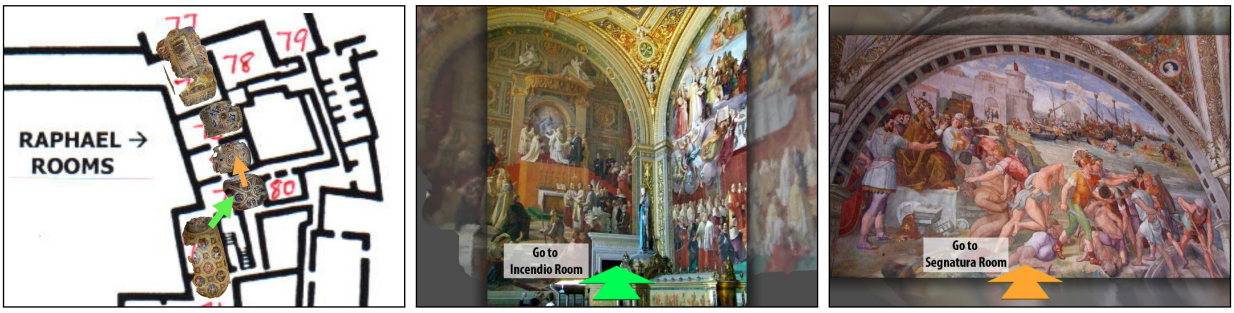


Figure 5.9: Sample paths of three tourists through the Vatican Museums represented as blue arrows on the provided floor plan. Two of them visited the Raphael rooms (top-right corner of the map) before the Sistine Chapel (bottom-right corner of the map), as is the designed path of travel through the museums, whereas the other did not visit the Sistine Chapel.

map of the Vatican Museum dataset, the first three Raphael Rooms appear to have a 2:1 aspect ratio, although our 3D models indicate an aspect ratio closer to 1:1. By consulting other maps from different sources, we are able to determine that the aspect ratio of our models is actually correct, i.e., the map is wrong. Being able to register multiple maps together and detect these map inaccuracies is a promising direction for future work.

5.5.2 Applications

The recovered reconstructions can be used to retrace a user visit to a large indoor site, like the Vatican Museums. By registering all photos of a user visit to the Vatican Museums to the registered 3D models and sorting them by their timestamp, we can recover its path through the museum. This information can be useful for visitors spatiotemporal behaviours in the Museum, like estimating how long visitors take to go through the Rafael rooms or what's the percentage of visitors skipping the Pio-Clementino Museum. Figure 5.9 shows the paths three different Flickr users took through the Vatican Museums. Note, however, that the path is traced directly on the 2D map and does not correspond to a real path a



(a) (b) (c)

Figure 5.10: Screenshots of our interactive visualization: (a) The annotated map is shown with the aligned 3D models rendered on top. When the user clicks on the model of the Hall of Immaculate Conception, the visualization flies into the room showing a photo taken in it (b). An arrow points to the location of the next room, the Incencio Room, and when clicked, the visualization flies the user to that room (c).

visitor could take, as the map depicts two floors and lacks annotations about staircases.

We also present a novel interactive web visualization tool of our indoor 3D reconstructions. We illustrate the interactions of the visualization tool in Figure 5.10, but we refer the reader to the supplemental video[1]. It features two navigation modes to explore the map and the reconstructed geometry. In map navigation mode, we allow common panning and zooming capabilities of the map. When you click on a room that has been assigned a 3D piece, the visualization automatically flies into the aligned 3D piece. You can navigate through the piece via an image-based rendering visualization, similar to the one in PhotoTourism [105]. Finally, when you look towards a neighbouring room, an arrow appears on the bottom of the screen pointing towards it. Clicking on the arrow makes the visualization transition between the two rooms, recreating the experience of moving from one room to another.

[1]Supplemental video available at the project website:
`http://grail.cs.washington.edu/projects/jigsaw3d/`

5.6 Conclusion

This chapter introduced the first system to reconstruct large indoor spaces of famous tourist sites from Internet photos via joint reasoning with map data and disconnected 3D pieces returned by Structure-from-Motion. We framed the problem as a 3D jigsaw puzzle and formulated an integer quadratic program with linear constraints that integrate cues over the pieces' position, scale, and orientation. We also introduced a novel crowd flow cue that measures how people travel through a site. Experiments on multiple sites showed consistently high precision for 3D model assignment and orientation relative to the input map, which allows for recovering tourists paths through these spaces and high quality interactions in the visualization tool. Our system works on popular tourist sites as it requires lots of images, text, and image metadata.

Chapter 6

CONCLUSIONS AND FUTURE WORK

In this thesis, I have proposed several ways to explore and understand the visual history of the world. The approaches and methods presented tackle this goal focusing at three different time scales, that range from millennia to a single day. At one of the spectrum is the written, recorded history of the world, as is carefully documented in books and reference texts. At the other end is the immediate history of our lives, including our visits to famous landmarks, that we carefully document by taking hundreds of photos, serving as breadcrumbs through these spaces that enable us to relive our experiences. In the middle, there is the visual history of the world in the online photo-sharing era, where we have access online to thousands of photos of every famous landmark in the world. Below, I summarize the contributions of my thesis towards visualizing and understanding the world's history in each of these time scales:

- The world's visual history in the last decade is captured by the millions of photos that tourists take and share online of the world's landmarks. I proposed a method in Chapter 2 that synthesizes time-lapse videos of these landmarks over multiple years from publicly available Internet photos. The method automatically discovered more than 10,000 time-lapse locations around the world from a database containing 86 million geotagged Picasa and Panoramio photos. The time-lapses depict the visual history of famous landmarks over the last decade starting around 2005, when Internet photo-sharing became popular. They show the many ways in which the world changes over time: from glaciers retreating, to growing city skylines or changing seasons. They sometimes reveal unexpected things, like the movement of the Charging Bull statue in New York, or the subtle weathering of statues over time. To generate even more compelling visualizations, I extended this approach in Chapter 3 to create 3D time-

lapses where the virtual camera moves continuously in space and time. The method is able to recreate cinematographic effects often used by professional filmmakers, like camera orbits or push movements, that add depth and dramatism to the time-lapses.

- To understand the history of a landmark beyond what is captured in Internet photos, Chapter 4 introduced the *3D Wikipedia*, a system that analyzes online text together with Internet photos to generate interactive visualizations that convey more efficiently the history of a place. The visualizations show on one side, a reference text, like a Wikipedia article of the site, and on the other, a faithful 3D reconstruction of the landmark computed from Internet photos. The system automatically recovers correspondences between the text and the 3D model by mining image and text co-occurrences across the Internet, and enables novel interactions for coordinated browsing of both the 3D model and the reference text. I demonstrated the 3D Wikipedia system and its novel interactions in five famous tourist sites, including the Pantheon in Rome, and the Pergamon Museum in Berlin.

- Finally, Chapter 5 presented the *3D Jigsaw Puzzle*, a system to reconstruct large indoor scenes from Internet photo collections. Recovering 3D models of indoor spaces is a key step towards unveiling the history of the millions of visits to places like the Vatican Museums, where geolocalization systems like GPS fail due to lack of signal. I frame the problem as solving a 3D jigsaw puzzle, where the pieces are the disconnected 3D reconstructions recovered by Structure-from-Motion methods, while leveraging a provided floor plan of the space. The system uses a novel crowd flow cue, that encodes how tourists travel through the space and is able to disambiguate the relative placement of neighbouring pieces. The 3D reconstructions obtained by solving this puzzle enable reconstructing the paths of tourists through these landmarks and novel photo-browsing experiences that convey the spatial context across neighbouring rooms.

6.1 Future Work

The results and visualizations presented in this thesis are just a few first steps towards fully visualizing the world's history. Below, I propose exciting directions of future work towards this goal.

6.1.1 Time-lapse of the World

The Internet time-lapses presented in Chapter 2 capture how the world's most famous landmarks are changing visually in 3D for many landmarks. However, the approach is limited to the places with very dense coverage, as it needs many photos, about 500, to average out the variability of Internet photos and create photorealistic time-lapses. Reducing the number of photos needed would increase the number of time-lapse locations manyfold, but requires more sophicasted modeling of the appearance of the scene in each input photo. Another avenue of future work is creating collaborative systems for people to upload their personal photos of landmarks, and generate time-lapse videos depicting the history of those landmarks over multiple decades. In any case, the rate at which photos are captured and uploaded to the Internet is ever increasing, and thus, these photos will provide a much denser coverage of the world's changes that are happening today compared to a decade ago.

In recent years, 360 degree photography has gained popularity with cameras like Ricoh Theta [88], and smartphone apps that stitch 360 panoramas, like Microsoft's Photosynth [84] and Google's PhotoSphere [44]. As more 360 photos are shared online, it opens the possibility of generating more immersive time-lapse visualizations, that cover the whole field of view from a vantage point. An example of a 360 time-lapse is shown in Figure 6.1, that is generated using 360 degree panoramic photos downloaded from the Space Needle's webcam [80] in Seattle over a period of 5 months. In this case, the webcam is static and the photos are aligned, so we only performed appearance regularization over the sequence. The result can be seen as a 360 video, or, ideally, in a more immersive head-mounted display. Furthermore, the method described in Chapter 2 can be extended to generate 360 panoramic time-lapses

(a) Screenshot of Space Needle Webcam (b) Spherical panorama time-lapse

Figure 6.1: Using the 360 degree panoramas available from the Space Needle Webcam [80] (a), we compute a 360 degree time-lapse video (b) of the evolution of the city of Seattle over a period of 5 months.

in places where there is enough imagery looking in all directions. By changing the output camera model to a spherical camera, the method can compute a spherical depthmap, and warp tourist photos to a virtual spherical camera and generate then a 360 degree panorama time-lapse. Figure 6.2 shows compares the field of view of a prototype spherical time-lapse video with the one computed for a canonical image of the scene.

Moreover, another worthy goal is generating Hollywood-quality time-lapse videos, that could be used, for example, in nature documentaries. In principle, this is possible, as many available photos have a high-resolution comparable to 4K video. However, stabilizing for the different viewpoints and the variability in appearance in real-world scenes, that contain plants and natural elements, to create photo-realistic high-resolution videos is still an open problem, that will require breakthroughs in 3D modeling and image-based rendering.

Another direction of future work is to incorporate the temporal aspects captured by the Internet photos into large-scale 3D models of the world, like the ones from Apple Maps [8], HERE [50], and Google Maps [42], that are built using satellite images, aerial, and street level imagery. Some of these models, like Street View Time Machine [43], or Google Earth

(a) Time-lapse result from Figure 2.14(d) (b) Spherical panorama time-lapse

Figure 6.2: Comparison between the fields of view of a perspective time-lapse and a prototype spherical time-lapse video of Mammoth Hot Springs in Yellowstone National Park. (a) frame of a time-lapse video computed from a reference photo's viewpoint, as in Chapter 2. (b) frame of a spherical time-lapse video using photos of the scene taken in all directions, that visualizes a larger field of view of 180×60 degrees.

Historical Imagery [41], already exhibit temporal information by displaying reconstructions of the world computed from imagery at different points in time, similar to how individual photos visualize change in a rephotography sequence. However, Internet photos provide a much more dense sampling of the appearance and temporal changes, and have the potential to enable a living 3D model of the world, that changes minute to minute due to weather, traffic, etc. Such live 3D representation of the world would allow scientists to track trends as they happen in the world in real-time, from weather prediction to flowering patterns, at an unprecedented scale.

6.1.2 Towards the 3D Wikipedia

The *3D Wikipedia* system presented in Chapter 4 is a first step towards automatically building a visual encyclopedia of the world. However, the system has several limitations. For example, when an object is mentioned in the reference text multiple times, the system is not able to discern what is the most useful excerpt to show to the user. This is particularly

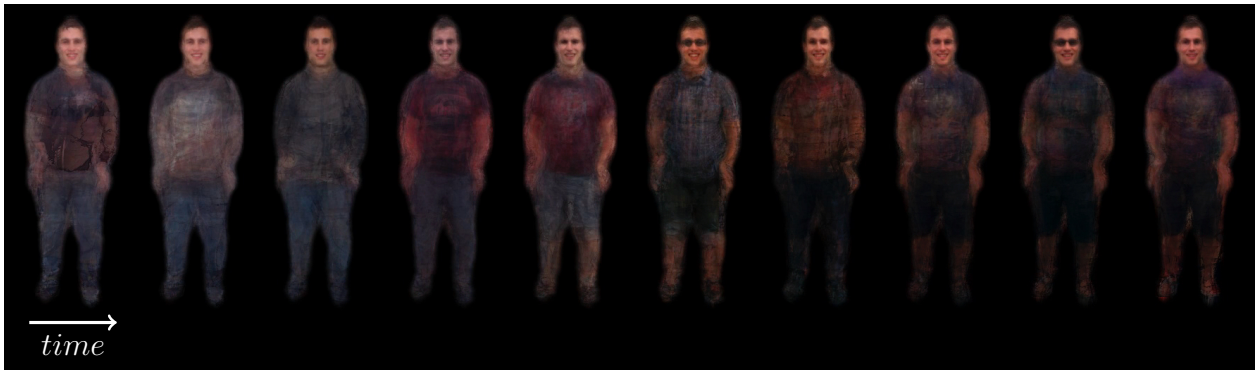


Figure 6.3: Sample frames of a proposed person time-lapse video, computed automatically from a personal photo collection. Note the subject's different appearances, like wearing a coat, short pants, shirts or sunglasses in some of frames.

confusing when analyzing multiple texts at the same time. Understanding multiple sources of text, and producing a single, comprehensive representation of the history of a place is a promising direction for future work.

Improving the quality and extent of the 3D reconstructions will also expand the applicability of the *3D Wikipedia*. The *3D Jigsaw Puzzle* presented in Chapter 5 is a step towards this goal, and enables reconstructing large indoor spaces from Internet photos using a provided floor plan of the space. A next step is to register these interior 3D models together with outdoors 3D models of the sites, that would enable fly-throughs animations from bird's eye view of cities into the rooms of a museum, or novel cutaway visualizations, that display, at the same time, the interior and exterior of a building and are popular in travel guides.

6.1.3 People's Visual History

The time-lapse techniques presented in Chapters 2 and 3 are limited to visualizing the changes in mostly static, rigid scenes, like famous landmarks. Extending these techniques to visualize changes in non-rigid subjects is a promising direction for future work. Indeed, most photos we take are of people, whose face and body pose are highly non-rigid. Picasa

Figure 6.4: Sample frames of a proposed family time-lapse video, computed automatically from a personal photo collection. Note the two children, that they appear after birth in the family portrait, and grow old over time. Used with permission of Ira Kemelmacher-Shlizerman.

Face Movies [59] visualize the aging of a person by showing photos through time, where the face and expression are stabilized. The results are compelling and remind to rephotography techniques, where instead the focus is only on the face of the person. However, Face Movies fail to create the illusion of time flowing continuously, that makes time-lapse videos so compelling. Furthermore, they only stabilize the person's face, making understanding changes in the person's body difficult. This would be very interesting, especially when visualizing teenagers' growth spurts, or changes in weight.

Figure 6.3 shows frames of a prototype time-lapse video of the author generated from his personal photo collection, that provides a glimpse to the underlying changes in the subject's appearance. The prototype was generated by first detecting photos of the subject in the photo collection using out-of-the-box face recognition systems, like the ones available in Facebook, Picasa, or Google Photos. Next, the subject was automatically segmented in each photo using a semantic segmentation approach [124]. To stabilize the pose throughout the sequence, the subject's pose was estimated using [20], and then the segmented photo is warped so the joint locations of the estimated pose coincide with the average joint locations for the whole sequence. Finally, the appearance of the sequence is regularized using the

technique described in Section 2.5. Another compelling application would be creating family portraits that continuously evolve over time, showing how children grow, and their hairstyles, facial features, and clothing change over time. A prototype of such time-lapse family portrait is shown in Figure 6.4, where a couple is at the top and their two children appear in the family portrait after their birth.

Finally, another exciting area for future work is to understand how society changes by exploring publicly available photos to model and visualize the population of certain areas over time. This would be very interesting in neighborhoods that have changed a lot over short periods of time, like the Mission District in San Francisco. The visualizations would show, for example, how the different demographic groups expand and shrink and people's fashions change over time. Creating such visualizations would be quite challenging, as no single photo is able to capture a complete picture of the people living in a neighborhood, and the demographic information has to be extracted from a large dataset of photos.

BIBLIOGRAPHY

[1] Austin Abrams, Kylia Miskell, and Robert Pless. The episolar constraint: Monocular shape from shadow correspondence. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2013.

[2] Jens Ackermann, Fabian Langguth, Simon Fuhrmann, and Michael Goesele. Photometric stereo for outdoor webcams. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2012.

[3] Sameer Agarwal, Yasutaka Furukawa, Noah Snavely, Ian Simon, Brian Curless, Steven M Seitz, and Richard Szeliski. Building rome in a day. *Communications of the ACM*, 2011.

[4] Sameer Agarwal, Keir Mierle, and Others. Ceres Solver. `http://ceres-solver.org`, 2012.

[5] Sameer Agarwal, Noah Snavely, Ian Simon, Steven M. Seitz, and Richard Szeliski. Building rome in a day. In *Computer Vision (ICCV), 2009 IEEE International Conference on*, 2009.

[6] Yushin Ahn and Jason E. Box. Glacier velocities from time-lapse photos: technique development and first results from the extreme ice survey (eis) in greenland. *Journal of Glaciology*, 2010.

[7] Apple. Apple iOS8 camera.

[8] Apple. Maps app for iOS. `http://www.apple.com/ios/maps`.

[9] Soonmin Bae, Aseem Agarwala, and Frédo Durand. Computational rephotography. *ACM Trans. Graph.*, 2010.

[10] Kobus Barnard, Pinar Duygulu, Nando de Freitas, David Forsyth, David Blei, and Michael I. Jordan. Matching words and pictures. *Journal of Machine Learning Research*, 2003.

[11] Alberto Bemporad. Hybrid Toolbox - User's Guide, 2004. `http://cse.lab.imtlucca.it/~bemporad/hybrid/toolbox`.

[12] Eric P. Bennett and Leonard McMillan. Computational time-lapse video. In *ACM SIGGRAPH 2007 Papers*, 2007.

[13] Alexander C. Berg, Tamara L. Berg, Hal Daumé, Jesse Dodge, Amit Goyal, Xufeng Han, Alyssa Mensch, Margaret Mitchell, Aneesh Sood, Karl Stratos, and Kota Yamaguchi. Understanding and predicting importance in images. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2012.

[14] Incorporated Berlitz International. *Berlitz Rome Pocket Guide*. Berlitz Pocket Guides Series. Berlitz International, Incorporated, 2003.

[15] Alessandro Bissacco, Mark Cummins, Yuval Netzer, and Hartmut Neven. Photoocr: Reading text in uncontrolled conditions. In *Computer Vision (ICCV), 2013 IEEE International Conference on*, December 2013.

[16] Yuri Boykov, Olga Veksler, and Ramin Zabih. Fast approximate energy minimization via graph cuts. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 2001.

[17] Marcus A. Brubaker, Andreas Geiger, and Raquel Urtasun. Lost! Leveraging the crowd for probabilistic visual self-localization. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2013.

[18] Chris Buckley. Automatic query expansion using SMART : TREC 3. In *Proceedings of the third Text REtrieval Conference (TREC-3)*, 1995.

[19] Matthew Carlberg, George Chen, Jacky Chen, John Kua, and Avideh Zakhor. Indoor localization and visualization using a human-operated backpack system. In *Indoor Positioning and Indoor Navigation (IPIN), 2010 International Conference on*, 2010.

[20] Joao Carreira, Pulkit Agrawal, Katerina Fragkiadaki, and Jitendra Malik. Human pose estimation with iterative error feedback. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.

[21] Taeg Sang Cho, Shai Avidan, and William T. Freeman. The patch transform. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2010.

[22] Taeg Sang Cho, Shai Avidan, and William T. Freeman. A probabilistic image jigsaw puzzle solver. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2010.

[23] Ondrej Chum, James Philbin, Josef Sivic, Michael Isard, and Andrew Zisserman. Total recall: Automatic query expansion with a generative feature model for object retrieval. In *Computer Vision (ICCV), 2007 IEEE International Conference on*, 2007.

[24] Neil Collins. Trajan's column. `http://www.visual-arts-cork.com/antiquity/trajans-column.htm`.

[25] Timothee Cour, Benjamin Sapp, and Ben Taskar. Learning from partial labels. *Journal of Machine Learning Research*, 2011.

[26] David J. Crandall, Lars Backstrom, Daniel Huttenlocher, and Jon Kleinberg. Mapping the world's photos. In *Proceedings of the 18th International Conference on World Wide Web (WWW)*, 2009.

[27] Erik Demaine and Martin Demaine. Jigsaw puzzles, edge matching, and polyomino packing: Connections and complexity. *Graphs and Combinatorics*, 23, 2007.

[28] Earth Vision Institute. Extreme Ice Survey. `http://extremeicesurvey.org`, 2007.

[29] Harold E. Edgerton and James R. Killian Jr. *Flash! Seeing the Unseen by Ultra High-Speed Photography*. Hale, Cushman & Flint, 1939.

[30] Boris Epshtein, Eyal Ofek, and Yonatan Wexler. Detecting text in natural scenes with stroke width transform. *2013 IEEE Conference on Computer Vision and Pattern Recognition*, 2010.

[31] Mark Everingham, Luc Van Gool, Christopher K. I. Williams, John Winn, and Andrew Zisserman. The Pascal visual object classes (VOC) challenge. *International Journal of Computer Vision*, 2010.

[32] Ali Farhadi, Mohsen Hejrati, Mohammad Amin Sadeghi, Peter Young, Cyrus Rashtchian, Julia Hockenmaier, and David Forsyth. Every picture tells a story: Generating sentences from images. In *Proc. of European Conference on Computer Vision (ECCV)*, 2010.

[33] Jan-Michael Frahm, Pierre Fite-Georgel, David Gallup, Tim Johnson, Rahul Raguram, Changchang Wu, Yi-Hung Jen, Enrique Dunn, Brian Clipp, Svetlana Lazebnik, and Marc Pollefeys. Building rome on a cloudless day. *Proc. of European Conference on Computer Vision (ECCV)*, 2010.

[34] Yasutaka Furukawa, Brian Curless, Steven M. Seitz, and Richard Szeliski. Towards internet-scale multi-view stereo. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2010.

[35] Yasutaka Furukawa and Jean Ponce. Accurate, dense, and robust multi-view stereopsis. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 2010.

[36] Andrew Gallagher. Jigsaw puzzles with pieces of unknown orientation. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2012.

[37] Duncan Garwood and Abigail Blasi. *Lonely Planet Rome*. Lonely Planet Publications, 2013.

[38] Michael Goesele, Noah Snavely, Brian Curless, Hughes Hoppe, and Steven M. Seitz. Multi-view stereo for community photo collections. In *Computer Vision (ICCV), 2007 IEEE International Conference on*, 2007.

[39] Ian J. Goodfellow, Yaroslav Bulatov, Julian Ibarz, Sacha Arnoud, and Vinay Shet. Multi-digit number recognition from street view imagery using deep convolutional neural networks. *International Conference on Learning Representations*, 2014.

[40] Google. Google art project. `http://www.google.com/culturalinstitute/project/art-project`.

[41] Google. Google Earth historical Imagery. `http://www.google.com/earth/explore/showcase/historical.html`.

[42] Google. Google Maps. `http://maps.google.com`.

[43] Google. Go back in time with Street View. `https://googleblog.blogspot.com/2014/04/go-back-in-time-with-street-view.html`, 2014.

[44] Google. Google camera app for android, 2016.

[45] A. E. Harrison. Reoccupying unmarked camera stations for geological observations. *Geology*, 1974.

[46] Richard I. Hartley and Andrew Zisserman. *Multiple View Geometry in Computer Vision*. Cambridge University Press, second edition, 2004.

[47] Daniel Hauagge, Scott Wehrwein, Paul Upchurch, Kavita Bala, and Noah Snavely. Reasoning about photo collections using models of outdoor illumination. In *Proceedings of BMVC*, 2014.

[48] James Hays and Alexei A. Efros. IM2GPS: estimating geographic information from a single image. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2008.

[49] Jared Heinly, Johannes Lutz Schönberger, Enrique Dunn, and Jan-Michael Frahm. Reconstructing the World* in Six Days *(As Captured by the Yahoo 100 Million Image Dataset). In *Computer Vision and Pattern Recognition (CVPR)*, 2015.

[50] HERE. HERE Maps. `http://maps.here.com`.

[51] InfoTrends. Worldwide infomage capture forecast, 2014.

[52] Instagram. Hyperlapse app.

[53] Nathan Jacobs, Brian Bies, and Robert Pless. Using cloud shadows to infer scene structure and camera calibration. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2010.

[54] Evangelos Kalogerakis, Olga Vesselova, James Hays, Alexei A. Efros, and Aaron Hertzmann. Image sequence geolocation with human travel priors. In *Computer Vision (ICCV), 2009 IEEE International Conference on*, 2009.

[55] Ryan Kaminsky, Noah Snavely, Steven M. Seitz, and Richard Szeliski. Alignment of 3D point clouds to overhead images. In *Workshop on Internet Vision*, 2009.

[56] Sing Bing Kang and Richard Szeliski. Extracting view-dependent depth maps from a collection of images. *International Journal of Computer Vision*, 2004.

[57] Kevin Karsch, Mani Golparvar-Fard, and David Forsyth. ConstructAide: Analyzing and visualizing construction sites through photographs and building models. *ACM Trans. Graph.*, 2014.

[58] Michael Kazhdan, Matthew Bolitho, and Hugues Hoppe. Poisson surface reconstruction. In *Proceedings of the 4th Eurographics Symposium on Geometry Processing (SGP)*, 2006.

[59] Ira Kemelmacher-Shlizerman, Eli Shechtman, Rahul Garg, and Steven M. Seitz. Exploring photobios. In *ACM Transactions on Graphics (proceedings of SIGGRAPH)*, 2011.

[60] Ted Kinsman. The time-lapse photography FAQ: An introduction to time-lapse photography. `http://www.sciencephotography.com/how2do2.shtml`.

[61] Dan Klein and Christopher D. Manning. Accurate unlexicalized parsing. In *Proceedings of the 41st Meeting of the Association for Computational Linguistics*, 2003.

[62] Felix Klose, Oliver Wang, Jean-Charles Bazin, Marcus Magnor, and Alexander Sorkine-Hornung. Sampling based scene-space video processing. *ACM Trans. Graph.*, 2015.

[63] David Koller, Jennifer Trimble, Tina Najbjerg, Natasha Gelfand, and Marc Levoy. Fragments of the city: Stanford's digital forma urbis romae project. *J. Roman Archaeol. Suppl.*, 2006.

[64] Johannes Kopf, Michael F. Cohen, and Richard Szeliski. First-person hyper-lapse videos. *ACM Trans. Graph.*, 2014.

[65] Lubor Ladický, Paul Sturgess, Chris Russell, Sunando Sengupta, Yalin Bastanlar, William Clocksin, and Philip H. S. Torr. Joint optimization for object class segmentation and dense stereo reconstruction. *International Journal of Computer Vision*, 2012.

[66] Pierre-Yves Laffont, Zhile Ren, Xiaofeng Tao, Chao Qian, and James Hays. Transient attributes for high-level understanding and editing of outdoor scenes. *ACM Transactions on Graphics (proceedings of SIGGRAPH)*, 2014.

[67] Vincent Laforet. Time Lapse Intro: Part I. `http://blog.vincentlaforet.com/2013/04/27/time-lapse-intro-part-i`, 2013.

[68] Dustin Lang and David W. Hogg. Searching for comets on the world wide web: The orbit of 17p/holmes from the behavior of photographers. *The Astronomical Journal*, 2012.

[69] Ivan Laptev, Marcin Marszalek, Cordelia Schmid, and Benjamin Rozenfeld. Learning realistic human actions from movies. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2008.

[70] E. Scott Larsen, Philippos Mordohai, Marc Pollefeys, and Henry Fuchs. Temporally consistent reconstruction from multiple video streams using enhanced belief propagation. In *Computer Vision (ICCV), 2007 IEEE International Conference on*, 2007.

[71] Anat Levin and Richard Szeliski. Visual odometry and map correlation. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2004.

[72] David G. Lowe. Distinctive image features from scale-invariant keypoints. *International Journal of Computer Vision*, 2004.

[73] Etienne-Jules Marey. *La Méthode Graphique Dans Les Sciences Expérimentales Et Principalement En Physiologie Et En Médecine*. Paris: Masson, 1885.

[74] Ricardo Martin-Brualla, David Gallup, and Steven M. Seitz. 3d time-lapse reconstruction from internet photos. In *Computer Vision (ICCV), 2015 IEEE International Conference on*, 2015.

[75] Ricardo Martin-Brualla, David Gallup, and Steven M. Seitz. Time-lapse mining from internet photos. *ACM SIGGRAPH 2015 Papers*, 2015.

[76] Ricardo Martin-Brualla, Yanling He, Bryan C Russell, and Steven M Seitz. The 3d jigsaw puzzle: Mapping large indoor spaces. In *Proc. of European Conference on Computer Vision (ECCV)*, 2014.

[77] Kevin Matzen and Noah Snavely. Scene chronology. In *Proc. of European Conference on Computer Vision (ECCV)*, 2014.

[78] Margaret Mitchell, Xufeng Han, Jesse Dodge, Alyssa Mensch, Amit Goyal, Alex Berg, Kota Yamaguchi, Tamara Berg, Karl Stratos, and Hal Daumé, III. Midge: Generating image descriptions from computer vision detections. In *Proceedings of the 13th Conference of the European Chapter of the Association for Computational Linguistics (EACL)*, 2012.

[79] Eadweard Muybridge. *Complete Human and Animal Locomotion*. University of Pennsylvania, 1887.

[80] Space Needle. Space Needle PanoCam. `www.spaceneedle.com/webcam/`, 2015.

[81] Richard A. Newcombe, Steven J. Lovegrove, and Andrew J. Davison. DTAM: Dense tracking and mapping in real-time. In *Computer Vision (ICCV), 2011 IEEE International Conference on*, 2011.

[82] John Ott. Exploring the spectrum. *Psychological Science*, 1974.

[83] James Philbin, Ondrej Chum, Michael Isard, Josef Sivic, and Andrew Zisserman. Lost in quantization: Improving particular object retrieval in large scale image databases. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2008.

[84] Photosynth (Windows Live Labs). `http://labs.live.com/photosynth`.

[85] Carlo Pietrangeli, Andre Chastel, John Shearman, John W. OḾalley, Pierluigi de Vecchi, Michael Hirst, Fabrizio Mancinelli, Gianluigi Colalucci, and Franco Bernabei. The sistine chapel: The art, the history, the restoration. *The Sixteenth Century Journal*, 1988.

[86] Rahul Raguram, Changchang Wu, Jan-Michael Frahm, and Svetlana Lazebnik. Modeling and recognition of landmark image collections using iconic scene graphs. *International Journal of Computer Vision*, 2011.

[87] Xiaofeng Ren, Liefeng Bo, and Dieter Fox. RGB-(D) Scene labeling: Features and algorithms. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2012.

[88] Ricoh. Ricoh Theta 360. `https://theta360.com`, 2016.

[89] Garry F. Rogers, Harold E. Malde, and Raymond M. Turner. *Bibliography of repeat photography for evaluating landscape change*. University of Utah Press, 1984.

[90] Michael Rubinstein, Ce Liu, Peter Sand, Frédo Durand, and Willaim T. Freeman. Motion denoising with application to time-lapse photography. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2011.

[91] B. C. Russell, R. Martin-Brualla, D. J. Butler, S. M. Seitz, and L. Zettlemoyer. 3D Wikipedia: Using online text to automatically label and navigate reconstructed geometry. *ACM Transactions on Graphics (SIGGRAPH Asia 2013)*, 2013.

[92] 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*, 2008.

[93] Gerard Salton and Chris Buckley. Improving retrieval performance by relevance feedback. *Journal of the American Society for Information Science*, 1999.

[94] Grant Schindler and Frank Dellaert. Probabilistic temporal inference on reconstructed 3d scenes. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2010.

[95] Grant Schindler, Frank Dellaert, and Sing Bing Kang. Inferring temporal order of images from 3d structure. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2007.

[96] Steven M. Seitz, Brian Curless, James Diebel, Daniel Scharstein, and Richard Szeliski. A comparison and evaluation of multi-view stereo reconstruction algorithms. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2006.

[97] Qi Shan, Riley Adams, Brian Curless, Yasutaka Furukawa, and Steven M. Seitz. The visual Turing test for scene reconstruction. In *Joint 3DIM/3DPVT Conference (3DV)*, 2013.

[98] Yichang Shih, Sylvain Paris, Frédo Durand, and William T. Freeman. Data-driven hallucination of different times of day from a single outdoor photo. *ACM Trans. Graph.*, 2013.

[99] Nathan Silberman, Derek Hoiem, Pushmeet Kohli, and Rob Fergus. Indoor segmentation and support inference from RGBD images. In *Proc. of European Conference on Computer Vision (ECCV)*, 2012.

[100] Ian Simon. *Scene Understanding Using Internet Photo Collections*. PhD thesis, University of Washington, 2010.

[101] Ian Simon and Steven M. Seitz. Scene segmentation using the wisdom of crowds. In *Proc. of European Conference on Computer Vision (ECCV)*, 2008.

[102] Ian Simon, Noah Snavely, and Steven M. Seitz. Scene summarization for online image collections. In *Computer Vision (ICCV), 2011 IEEE International Conference on*, 2007.

[103] Josef Sivic and Andrew Zisserman. Video Google: A text retrieval approach to object matching in videos. In *Computer Vision (ICCV), 2003 IEEE International Conference on*, 2003.

[104] Noah Snavely, Rahul Garg, Steven M. Seitz, and Richard Szeliski. Finding paths through the world's photos. *ACM Transactions on Graphics (Proceedings of SIGGRAPH 2008)*, 2008.

[105] Noah Snavely, Steven M. Seitz, and Richard Szeliski. Photo tourism: Exploring photo collections in 3d. In *ACM Transactions on Graphics (proceedings of SIGGRAPH)*, 2006.

[106] Noah Snavely, Steven M. Seitz, and Richard Szeliski. Modeling the world from Internet photo collections. *International Journal of Computer Vision*, 2008.

[107] Stop words list. `http://norm.al/2009/04/14/list-of-english-stop-words/`.

[108] Kalyan Sunkavalli, Wojciech Matusik, Hanspeter Pfister, and Szymon Rusinkiewicz. Factored time-lapse video. In *ACM SIGGRAPH 2007 Papers*, 2007.

[109] Frederik A. Talbot. *Moving pictures : how they are made and worked*. J. B. Lippincott Co., 1914.

[110] Alexandru Telea. An image inpainting technique based on the fast marching method. *Journal of Graphics Tools*, 2004.

[111] tesseract-ocr. `https://code.google.com/p/tesseract-ocr/`.

[112] Andreas Velten, Di Wu, Adrian Jarabo, Belen Masia, Christopher Barsi, Chinmaya Joshi, Everett Lawson, Moungi Bawendi, Diego Gutierrez, and Ramesh Raskar. Femto-photography: Capturing and visualizing the propagation of light. *ACM Trans. Graph.*, 2013.

[113] Chun-Po Wang, Kyle Wilson, and Noah Snavely. Accurate georegistration of point clouds using geographic data. In *3DV*, 2013.

[114] Gunther Wegner. LR-Timelapse. `http://lrtimelapse.com/`.

[115] Ethan Z. Welty, Timothy C. Bartholomaus, Shad O'Neel, and W. Tad Pfeffer. Cameras as clocks. *Journal of Glaciology*, 2013.

[116] Changchang Wu. SiftGPU: A GPU implementation of scale invaraint feature transform (SIFT). `http://cs.unc.edu/~ccwu/siftgpu`.

[117] Changchang Wu. VisualSFM - a visual structure from motion system. `http://ccwu.me/vsfm/`, 2011.

[118] Changchang Wu, Sameer Agarwal, Brian Curless, and Steven M. Seitz. Multicore bundle adjustment. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2011.

[119] Jianxiong Xiao and Yasutaka Furukawa. Reconstructing the world's museums. In *Proceedings of the 12th European Conference on Computer Vision*, 2012.

[120] Jianxiong Xiao, Andrew Owens, and Antonio Torralba. SUN3D: A database of big spaces reconstructed using SfM and object labels. In *Computer Vision (ICCV), 2013 IEEE International Conference on*, 2013.

[121] Guofeng Zhang, Jiaya Jia, Tien-Tsin Wong, and Hujun Bao. Consistent depth maps recovery from a video sequence. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 2009.

[122] Li Zhang, Brian Curless, and Steven M. Seitz. Spacetime stereo: shape recovery for dynamic scenes. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2003.

[123] Ke Colin Zheng, Alex Colburn, Aseem Agarwala, Maneesh Agrawala, David Salesin, Brian Curless, and Michael F. Cohen. Parallax photography: Creating 3d cinematic effects from stills. In *Proceedings of Graphics Interface 2009*, 2009.

[124] Shuai Zheng, Sadeep Jayasumana, Bernardino Romera-Paredes, Vibhav Vineet, Zhizhong Su, Dalong Du, Chang Huang, and Philip H. S. Torr. Conditional random fields as recurrent neural networks. *Computer Vision (ICCV), 2015 IEEE International Conference on*, 2015.

VITA

Ricardo Martin Brualla grew up in Madrid, Spain and received his B. Sc. in Mathematics and his B. Tech. in Computer Engineering from the Universitat Politecnica de Catalunya, Barcelona, Spain in 2009 and 2011. In 2009 he received the "La Caixa" Fellowship for Graduate Studies in the United States. In 2010 he joined the Ph.D. program at the Computer Science Department at University of Washington, where he was advised by Professor Steve M. Seitz. During his Ph.D., he did summer internships at Microsoft Research in Redmond, Wahshington, and Google in Seattle, Washington and Mountain View, California. In 2016, he received his Doctor in Philosophy degree.