Digital Video Projection for Interactive Entertainment

Naoto Hieda

Master of Engineering

Department of Electrical and Computer Engineering

McGill University

Montreal,Quebec

2015-08-14

A thesis submitted to McGill University in partial fulfillment of the requirements for the degree of Masters in Engineering.

ACKNOWLEDGEMENTS

First of all, I wish to express my most sincere gratitude and appreciation to my supervisor, Dr. Jeremy Cooperstock, for directing the research project and also giving valuable comments on the writing and presentation. I would like to thank all the current and past lab members, especially Dalia El-Shimy, Jan Anlauff, and Jeff Blum for overall advice on research; Meng Xie for advice on camera calibration; and Dr. Alexander Eichhorn for software development. I am also grateful to Dr. Samuel Audet for his valuable suggestions on computer vision. I would like to thank Dr. Yuki Hashimoto and Dr. Akihiko Shirai for valuable advice during the exhibition in Japan. My sincere thanks also goes to Dr. Judith Doyle, Ms. Marcella França and Mr. Daniel Biléu for their creative ideas. I gratefully thank Dr. Carolyn Samuel and Dr. Richard Cooper for their valuable courses on English writing as well as proofreading of the thesis draft. Last but not the least, I thank my parents for their support. This research was supported by the Networks of Centres of Excellence in Graphics, Animation, and New Media.

ABSTRACT

Digital projection technology allows for effective and entertaining spatial augmented reality applications. Leveraging the capabilities of state-of-the-art motion capture or commodity depth sensors to determine the 3D position and pose of objects in real time, it is possible to project dynamic graphical content on arbitrary surfaces. In this thesis, we explore computer vision techniques including projector-camera calibration and 3D surface reconstruction to accurately map contents on a static surface, a dynamically moving rigid-body surface, and a human face. Quantitatively, the projection error is measured for both displaced and moving rigid-body objects. The accuracy of projection on a displaced object is within 2 pixels in 71% of the extent of the measured points in 320 mm $\times$ 400 mm working area. The system's response time to object movement is dictated primarily by that of the latency of the acquisition and display devices used, and a prediction filter is implemented for delay compensation. As an application of such a dynamic projection mapping system, we studied digital facial augmentation whereby participants can have the experience of "painting" on someone's face, or even on their own, by observing the projection in a mirror.

ABRÉGÉ

La technologie de vidéoprojection numérique nous permet de créer efficacement des applications amusantes de réalité spaciale augmentée. En exploitant les capacités de capture de mouvements de pointe ou de scanneurs 3-D il est possible de déterminer la position et l'orientation des objets en temps réel. Cela nous permet de projeter du contenu graphique dynamique sur des surfaces variées. Dans cette thèse nous explorerons les techniques de vision par ordinateur incluant la calibration de caméra-vidéoprojecteur et la numérisation de surfaces 3-D dans le but de projeter du contenu sur une surface statique, un modèle du solide indéformable et un visage humain. Quantitativement, l'erreur de projection est mesuré pour les objets déplacés ainsi que pour les modèles du solide indéformable. L'erreur de précision de la projection sur un objet déplacé est à l'intérieur de 2 pixels dans 71% des points mesurés dans un espace de travail de 320 mm $\times$ 400 mm. Le temps de réponse du système par rapport aux objets en mouvement est causé par la latence du scanneur et du vidéoprojecteur utilisés. Dans ce cas nous implantons un filtre pour compenser le délai. Pour démontrer l'application de cette vidéoprojection de mapping dynamique, nous avons étudié l'augmentation numérique du visage où les participants réalisaient l'expérience de peindre sur leur visage ou celui de quelqu'un d'autre en observant la projection dans un miroir.

TABLE OF CONTENTS

ACKNOWLEDGEMENTS ii

ABSTRACT iii

ABRÉGÉ iv

LIST OF FIGURES vii

1 Introduction 1

1.1 Thesis Outline 5
1.2 Contibution of the Thesis 5

2 Literature Review 7

2.1 Camera Calibration 7
2.2 Structured Lighting 9
2.3 Object Tracking 13
2.4 Video Projection Systems 15

3 Static Projection Mapping 19

3.1 Methodology 19
3.1.1 3D Geometry Acquisition and Camera Calibration 20
3.1.2 Structured Lighting 21
3.1.3 Self Camera Calibration 22
3.1.4 Radial Fundamental Matrix 24
3.1.5 Texture Mapping 25
3.1.6 OpenFrameworks and Implementation 26
3.2 Conclusions 30

4 Interactive Projection Mapping Using Motion Capture 32

4.1 Methodology 32
4.1.1 Motion Capture 34

4.1.2 Marker Extraction 37
4.1.3 Mesh Segmentation 37
4.1.4 Calibration of Motion Capture and Projector 38
4.1.5 Video Projection 39
4.2 Quantitative Results 41
4.3 Conclusions 44

5 Interactive Projection Mapping Using a Depth Camera 47

5.1 Implementing a Facial Augmentation System 48
5.1.1 Calibrating the Hardware 50
5.1.2 Depth Camera Marker Tracking 51
5.1.3 Prediction Filter 53
5.1.4 User Interaction 54
5.2 Enhancements for Effective Interactive Entertainment 56
5.3 Conclusions 62

6 Conclusions 64

6.1 Future Work 65

References 68

LIST OF FIGURES

Figure page

1–1 Demonstration at Graphics, Animation and New Media Conference 2014. 3

1–2 Demonstration at International Collegiate Virtual Reality Contest 2014. 3

2–1 Structured light patterns proposed by (a) Posdamer et al.; (b) Horn et al.; (c) Gühring; (d) De Bruijn; (e) Salvi et al.; (f) Morano et al.; (g) Sato. The figure is taken from Salvi et al. [41]. 11

3–1 Time-multiplexed gray-coded structured light projected on the scene to obtain initial 3D geometry. Top row: first 8 vertical patterns. Bottom row: first 8 horizontal patterns. 21

3–2 Phase-shifted sinusoidal structured light projected on the scene. The first 4 images show vertical patterns, and the last 4 images show horizontal patterns, both at a wavelength of 8 pixels. The images as shown here are cropped to improve visibility of the patterns. . . 22

3–3 A diagram of the vertical center. 28

3–4 Projector-camera experimental setup. 30

4–1 Experimental setup. The projector on the right, the three OptiTrack cameras on the left of the image, and the Point Grey camera in front of them are used for the kinetic video projection. 33

4–2 Markers for the kinetic video projection system. A retro-reflective sheet cut into a disk, which can be detected by IR active cameras, is surrounded by a black border for blob extraction from a color camera image. 35

4–3 A screenshot of the marker extraction result is shown in (a). The threshold is determined ad-hoc for the best result. In (b), a reconstructed mesh with texture mapped is shown. 36

4–4 Projection of a texture on a dynamic object. 39

4–5 Illustration of a sequence of kinetic video projection. 40

4–6 For RMSE estimation using homography, first, (a) checkerboard corners are detected from a camera image. (b) Then, a circle pattern is detected from a warped image, and the RMSE is calculated from the circle center displacement. 41

4–7 Projection RMSE plotted against the XZ plane. The measured RMSEs (in mm) are scaled by the pixel dimension, which varied between 0.43 and 0.63 mm depending on depth. The gray areas indicate RMSE cannot be measured due to the limitation of the projector field of view. 43

4–8 Projection RMSE (in mm) on a moving object, as a function of object speed. The RMSE is very close to the amount of object motion during the minimum estimated inherent system latency, and unsurprisingly, is positionally biased behind the object in the direction of its motion. 45

5–1 Side view of the LED-marker-based prototype. 48

5–2 Tablet interface. On the left half of the image, a participant with the goggles is shown. On the right half, stickers are displayed along with pen color selection buttons. 49

5–3 Photos of the improved face projection system. In Figure 5–3a and 5–3b, finger tracking methods are integrated to possibly assist drawing on the own face. In Figure 5–3c and 5–3d, 2D physics engines are applied to render virtual objects. 52

5–4 Specular reflection shading. Highlights can be seen on the cheek and jaw, and move across the face according to the head orientation. . . 55

5–5 Metal reflection shading. Similar to the specular reflection example, surface areas whose normals are parallel to the incident direction of light, i.e., similar to the optical axis of the camera that captured the photos above, are highlighted. 55

5–6 Parallax rendering. The projected lines are visible through a mirror, and can be interpreted as parallel line segments extending from the face. 56

5–7 A photo taken during the demonstration at the International Collegiate Virtual Reality Contest 2014. 57

5–8 An example of a texture-mapped mesh with overlaid wireframe. . . . 58

5–9 A flowchart of the system. 60

5–10 Fingertip detection. 61

5–11 A photo taken during the demonstration at Laval Virtual 2015. 62

CHAPTER 1
Introduction

Video projection technology can be used not only to generate contents on a planar screen, but also, to produce sophisticated theatre stage or art installation lighting. In contrast to conventional lighting, such as spotlights, projection can achieve the same results without mechanical motion, color filters, or masking foils, thereby reducing the complexity of operation. For example, the Theatre at the University of British Columbia (UBC) is employing multiple projectors to produce visual effects on performing artists. Other groups using projection include Cirque du Soleil with Michel Lemieux and Victor Pilon, who presented Delirium [11], a show featuring extensive semi-transparent screens to overlay projection on stage performance as a Pepper's ghost effect. A technique for mapping textures or videos to specific surfaces for augmenting a physical environment is referred to as projection mapping; given textures or videos to project, they are prewarped by, for example, a homography for a planar quadrangle, a Bezier surface model for a polynomial surface [39] or a 2D parametric domain for an arbitrary surface. A homography transformation is a linear mapping that requires four corners in a physical space corresponding to the corners of a rectangle. Points within the rectangle are interpolated from the four points. Despite its simplicity, such a homography is often sufficient for projection mapping on man-made objects. Commercial software for video generation such as vvvv [51], MadMapper [30] and

X-Agora [52] is bundled with a homography warping feature. Notably, MadMapper has a gray-coded structured lighting feature to facilitate video mapping using a projector-camera pair; however, this feature is limited to warp a camera image to generate an image which is supposed to be taken from the projector perspective. Then, an artist has to paint on the image without having depth cues of the geometry.

By exploiting the structured lighting technique and by introducing camera calibration, the 3D geometry of the scene can be reconstructed in addition to the pixel correspondences generated by such commercial products. 3D information significantly increases the potential of projection mapping installations; for example, depth-sensitive effects applied to video contents, texture color varying according to depth, or video projection on a moving object by means of real-time tracking. The original motivation for the work presented in this thesis was to develop a set of tools that could support efficient video mapping onto an arbitrary stage performance area by acquiring a 3D geometry model of the scene automatically to avoid the time-consuming manual mapping procedures. Besides stage lighting, we discovered that the system can also be applied to support interactive art installations. We built installations that track participants and project video contents generated by users, and successfully demonstrated at three venues.

In addition to providing a homography transformation for multiple projectors, a 3D geometry of the scene enriches video projection applications. First, the 3D geometry can be useful for optical illusions often preferred in projection mapping

Figure 1–1: Demonstration at Graphics, Animation and New Media Conference 2014.

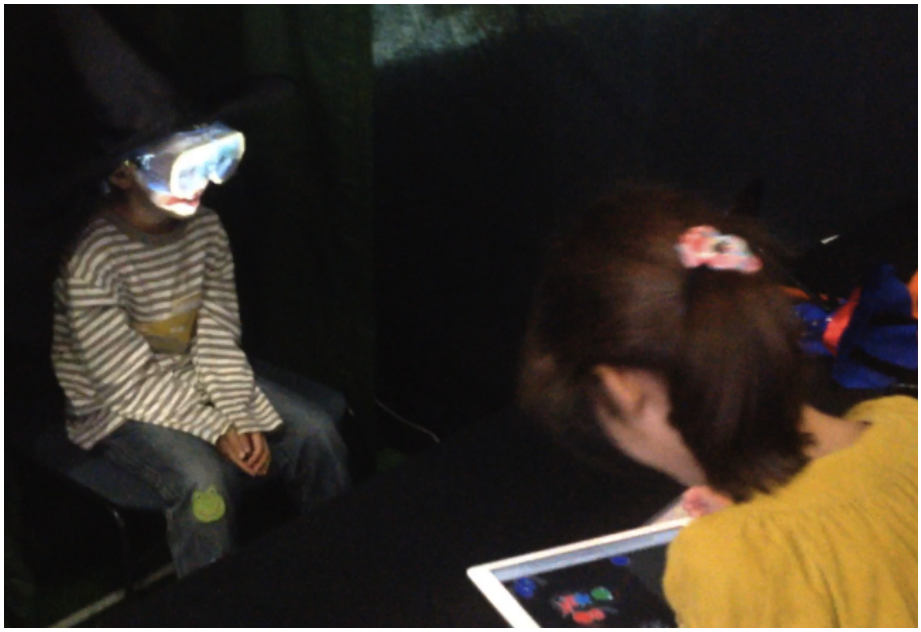


Figure 1–2: Demonstration at International Collegiate Virtual Reality Contest 2014.

installations such as re-lighting and 3D illusion, both seen in projection mapping on the Hague City Hall [47]. Second, projection on a moving, arbitrary surface is made possible as long as its motion is controlled or tracked in real-time; for example, as demonstrated by the projection on a mechanically controlled object in the video of Bot & Dolly's Box [10] or tracked human faces [7] [48]. In the beautifully choreographed Bot & Dolly's Box, the motion of the mechanically controlled projection surfaces is scripted, and thus accurate projection is made possible. Even though the object is not controlled in such a method, when the object position and orientation are observed in real-time, a texture can be correctly distorted by a renderer and projected on the physical surface. For example, to track the position and orientation, a motion capture system that consists of retro-reflective markers and high-speed active infrared (IR) cameras can be integrated. Such high-speed cameras and depth-sensing cameras gave rise to video projection applications that map contents on the human body. Even though projection on a face is particularly challenging because of its uniqueness to each individual and deformability, several attempts are made; for example, Bell's music video "Chase No Face" [7] achieves projection by facial recognition and a depth camera, and a video by Oskar and Gaspar [12] projects on a fixed face. A recent demonstration of facial projection is performed by using motion capture with special makeup to print IR reflective markers on a face [48]. However, these applications require time-consuming individual calibration procedures and may not be robust to rapid motion. We avoid the problems of calibration and stability by a face mask or goggles and a motion capture system with which users

can instantly participate in facial projection. As an application of an object-tracking projection system, interactive video projection on a human body is proposed. While interactive video projection systems have become common in art installations, such as MobiSpray [43], we integrate such an idea of interactive projection to real-time tracking to engage participants.

1.1 Thesis Outline

The remainder of the thesis is organized as follows. First, preceding research on projector-camera systems is reviewed (Chapter 2). Next, projector and color camera calibration, 3D reconstruction of the physical scene, and texture mapping on a planar surface are explained for projection on a static scene (Chapter 3). Then, using rigid-body object tracking, projection mapping on a moving object is described, and its accuracy is quantitatively evaluated (Chapter 4). Finally, an interactive, facial projection mapping system that uses a depth camera for marker-based tracking or markerless face tracking is described (Chapter 5).

1.2 Contibution of the Thesis

The major contribution in this thesis is an evaluation method for projector calibration. Camera calibration is a process that estimates camera parameters (e.g., a focal length and a camera position) from a 3D geometry of the target object (e.g., a checkerboard). Although a calibration algorithm minimizes the distance between a measured point (e.g., a checkerboard corner) and the corresponding 3D point projected on the image plane, they do not align perfectly due to the camera parameter errors. This distance, known as the reprojection error [21], is usually employed to evaluate the quality of camera calibration. The

projector calibration is evaluated by the same method in previous systems [5, 53]. However, for a projection mapping system, this error does not represent the displacement of the pixels projected on a physical surface because the reconstructed 3D geometry may also contain errors. Thus, we make use of the root mean squared error (RMSE) of the projection position, observed by an external camera, which provides the ground truth of the 3D geometry (Chapter 4).

When the projection accuracy is measured on a moving object that can be tracked in realtime, the RMSE of the projection position represents the object displacement during the time interval from position acquisition to projection update. Therefore, the system latency can be evaluated. Given the latency, a linear prediction filter can be implemented, which has a critical role for certain applications. For example, we developed a facial projection mapping system, whose video content must avoid projection on eyes by exploiting the prediction filter (Chapter 5).

CHAPTER 2
Literature Review

In this chapter, the background of theories and methods related to video projection systems are reviewed. First, camera calibration, a computer vision approach to estimate camera and projector geometry is reviewed in Section 2.1. Next, structured lighting methods for acquiring 3D geometry using a calibrated projector-camera system are introduced in Section 2.2. Then, real-time object tracking methods including applications of structured lighting, which are essential to dynamic video projection, are explained in Section 2.3. Finally, in Section 2.4, video projection systems exploiting a projector-camera setup and object tracking from art installations, virtual reality and augmented reality applications are described.

2.1 Camera Calibration

In camera-projector systems, such as often found in augmented reality applications, a requisite initial step is that of calibration, i.e., solving for the intrinsic and extrinsic parameters of each device. Since a projector is assumed to follow the pinhole camera model, camera calibration methods can be applied to the projector. With the widespread calibration method introduced by Zhang [56], a checkerboard pattern in different poses is captured by a camera. Then, coordinates of the checkerboard intersections are extracted. Finally, intrinsic and extrinsic camera parameters are calculated by fitting these coordinates. This method can be

extended to calibration of projectors by projecting a checkerboard or circle pattern onto a flat white surface. By combining Zhang's method [56] and structured lighting (e.g., gray coding [41]), which finds the correspondences between projector and camera pixels, the 3D point cloud of the scene is reconstructed.

A downside of Zhang's camera calibration method is that manual intervention is required to change the pose of the checkerboard; what is worse, since this method can only calibrate one projector and one camera at the same time, the camera parameters must be chained through the projectors and cameras for a multiple projector-camera setup, which possibly accumulates calibration errors. To overcome these problems, several self-calibration algorithms, which produce camera parameters only from pixel correspondences, have been developed [53, 46]. For instance, an essential matrix decomposition technique for a projector-camera pair or a pair of two cameras computes the camera parameters as follows. First, from the pixel correspondences, the eight-point algorithm can be applied to compute a fundamental matrix. Here, we assume that the principal points of the projector and camera image planes are known, and the pixel dimensions are square. Then, the focal lengths can be estimated from the fundamental matrix and principal points [8], and consequently, the intrinsic matrices can be formed by the principal points and the estimated focal lengths. Next, the image coordinates of the fundamental matrix are normalized by the intrinsics, i.e., the essential matrix is calculated. Finally, applying singular value decomposition to the essential matrix gives the extrinsics [21]. Since focal length is prone to error, Yamazaki et al. [53] proposed applying a Levenberg-Marquardt fitting to all the camera parameters.

By exploiting essential matrix decomposition, a projector and a camera can be calibrated automatically.

Svoboda et al. [46] proposed another self-calibration method for calibrating three or more cameras. When pixel correspondences among the cameras are known, projection matrices and the 3D shape can be recovered by solving a tessellated projection equation without ambiguity using the projection matrix rank property and geometric constraints. This approach is called structure-from-motion (SFM). Similar to the essential matrix decomposition method, SFM can be extended to calibration of projectors, for which reliable and dense pixel correspondences can be determined by structured light. Consequently, adapting one of the self-calibration methods, digital video projection can be performed with little manual intervention.

2.2 Structured Lighting

Structured lighting is a method to find pixel correspondences between a projector-camera pair. One or more coded patterns as shown in Figure 2–1, for which codewords are assigned to its pixels or blobs, are projected on a scene, and then, these codewords are decoded from camera images. With binary or n-ary coding, codewords are uniquely assigned to each pixel, and therefore, there is a one-to-one correspondence between the camera and projector pixels since the codewords are multiplexed in the time domain, which requires that the scene remains static until all the patterns are projected [41]. The first binary coding method is proposed by Posdamer et al. [37], whereby a codeword is represented by two gray levels (black and white); thus, this method is called gray coding. Gray

coding is extended by Gühring [19] to add sub-pixel precision by integrating phase shifting and by Horn et al. [23] to increase gray levels to use n-ary codewords and to reduce the number of structured light patterns by $2\lfloor \log_2 n \rfloor - 2$, where n is the number of gray levels. Another coding method, neighborhood codification, requires a spatially coded single pattern so that the pixel correspondences can be recovered from a single camera image. Since this method does not require the scene to remain static between multiple frames, real-time 3D scanning is made possible although its resolution is limited because blobs are used instead of pixels to represent codewords. Neighborhood codification is further categorized into two strategies: single spatial axis codification (De Bruijn method [41]) and dual spatial axis codification (Salvi et al. [40] and Morano et al. [32]). With the De Bruijn method, codewords are encoded as slit patterns in a single direction, and Salvi et al. improved the robustness by adding perpendicular slits. Alternatively, Morano et al. used a matrix pattern with colored dots that are spatially encoded. Finally, direct codification methods exploit periodicity to assign a unique codeword to each pixel. Since this method is vulnerable to surface reflectance, Sato [42] proposed using several color patterns with linearly mapped hue values to eliminate the surface color.

A quantitative comparison of various structured light coding techniques is provided by Salvi et al. [41]. Seven techniques for 3D reconstruction, which are described above, are tested, and the depth reconstruction error and number of pixels reconstructed by each technique were measured, as shown in Table 2–1.

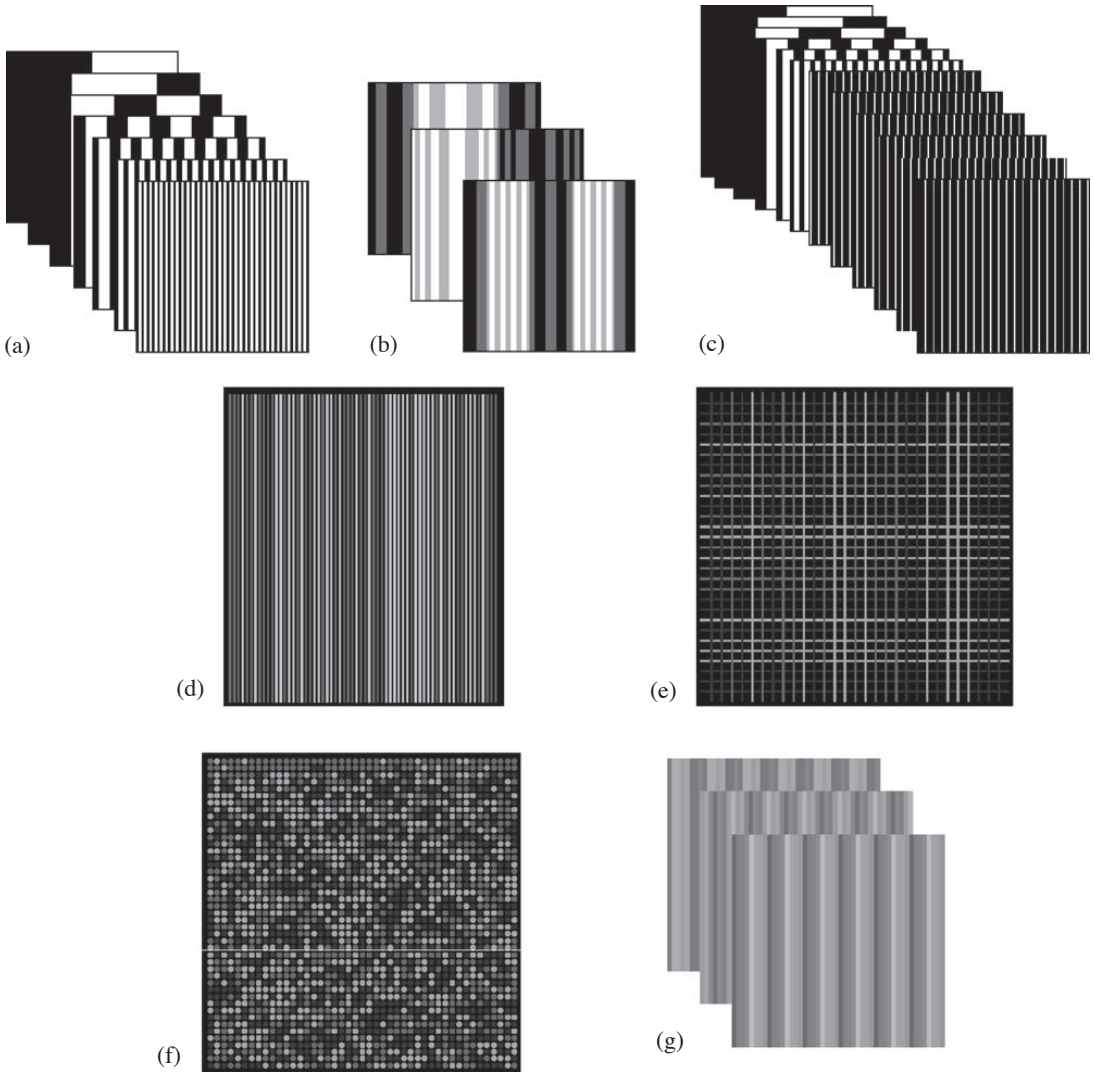


Figure 2–1: Structured light patterns proposed by (a) Posdamer et al.; (b) Horn et al.; (c) Gühring; (d) De Bruijn; (e) Salvi et al.; (f) Morano et al.; (g) Sato. The figure is taken from Salvi et al. [41].

Technique	Standard deviation (μm)	Number of 3D points	Resolution (%)	Number of patterns
Posdamer	37.6	25213	21.67	9
Horn	9.6	12988	11.17	5
Gühring	4.9	27214	23.38	14
De Bruijn	13.1	13899	11.94	1
Salvi	72.3	372	0.32	1
Morano	23.6	926	0.80	1
Sato	11.9	10204	8.77	3

Table 2–1: A quantitative comparison of structured lighting methods studied by Salvi et al. [41].

All seven methods were tested with the same projector (1024 $\times$ 768 pixels), camera (768 $\times$ 576 pixels) and scene. Comparing the various methods on the basis of standard deviation of the depth reconstruction errors, codification strategies with sub-pixel precision, i.e., the methods of Horn, Gühring, De Bruijn and Sato, achieved greater accuracy than did the other three methods, which only yield pixel-wise correspondence. Besides having the highest accuracy of the seven methods, Gühring's method achieved the highest resolution, i.e., density of 3D points, but at the cost of the largest number of projected patterns including gray code and phase-shifting sinusoidal patterns, thus imposing the greatest time requirements. Though Horn's n-ary coding and Güring's binary-coding methods yielded comparable accuracies, n-ary coding reduces the number of patterns while its resolution is less than binary coding due to the noise sensitivity.

It is worth noting that the results of Salvi et al. indicate that this method is not robust to depth discontinuities of the scene. De Bruijn and Sato's methods result in dense 3D points in certain cases as seen in Table 2–1; however, these

methods yield less resolution when the depth discontinuity is significant. Contrarily, the two-axis neighborhood codification techniques (Salvi and Morano) can produce stable precision and resolution since the resolution is intentionally limited in order to encode unique codewords as much as possible in a single pattern.

The first generation of Kinect, a widespread depth camera by Microsoft, achieves depth scanning based on neighborhood codification, which encodes codewords in a single pattern for real-time scanning (30 frames per second). The camera produces a dense depth image that recovers more than 90% of the image resolution [26] compared to the methods studied by Salvi et al., whose largest amount of points recovered is 23.38 % by Gühring's method. This density is owed to the Kinect's use of an IR laser emitter instead of an LCD projector. However, the depth resolution achieved by Kinect is 2 cm at most [24], and the average depth error is 500 μm, which is significantly worse than the methods shown in Table 2–1.

2.3 Object Tracking

In the following, we review the literature relevant to object tracking for augmented reality systems. Such systems often require tracking techniques to locate the objects of interest and overlay contents on the physical surface by projection, which is further explained in Section 2.4. We review tracking methods achieved either by structured lighting, which acquires object geometry, or by markers affixed to objects.

Structured Lighting Although structured lighting can be used for projector-camera calibration and 3D reconstruction to measure the geometry

for video projection (see Section 2.2 for details), one of its techniques, time-multiplexed structured light, can be used only for static objects. This technique is suitable for initial estimation of the scene, but the initial 3D model must be updated when the objects are moved. Another technique, one-shot structured light [41], as used by the Microsoft Kinect, can obtain the depth image in real-time and is thus able to update the 3D model. Nonetheless, with any structured lighting methods, the output is limited to a set of 3D points, which does not directly provide the information of the objects in the scene (e.g., the pose or label of an object). Such information is often desired for systems that warp video contents using the position and orientation of the target and accurately project on its surface. When structured-lighting scanning is used, the target pose is only acquired by fitting a pre-modeled 3D mesh to the scanned depth image using methods such as iterative closest point. However, this method is computationally expensive and ill-posed, requiring constraints on object motion and use of GPU acceleration to be tractable [33].

Marker Tracking Color cameras are suitable for object tracking in real-time applications, exploiting fiducials [39] or an optical model of the object surface [6] to estimate the relative pose between the projector and target; however, many existing systems project contents on planar or quadratic surfaces using homography or quadratic parameterization to warp the contents but not on complicated surface structure. The drawback of such systems is that tracking can be achieved only under ambient illumination to be visible by a color camera, although most video projection applications assume dark environments to compensate for the limited

luminance of projectors. To overcome this problem, motion capture systems are often used, which employ IR LEDs to illuminate retro-reflective markers attached to the target and observe them by three or more IR-sensitive cameras to triangulate the marker positions [48]. Similarly, structured lighting can be integrated to a marker tracking system by affixing photosensors to the target and locating the photosensors by decoding the received codewords [28] [27].

2.4 Video Projection Systems

Projectors are often used for immersive display environments for virtual reality systems since they can be scaled more easily than liquid-crystal displays (LCDs), which have fixed size. One of the early and successful demonstrations of this approach is the CAVE [15], which surrounds users by projection covering the walls. However, this required a dedicated physical environment, which made the system non-portable and inaccessible to consumers. Thus, numerous approaches are made to deploy a CAVE-like system in a room, which require keystone correction and projection blending techniques. For example, Garcia-Dorado et al. [18] used fiducials affixed to walls, projectors and a mechanically controlled camera. The camera detects the projection surface through gray-coded structured lighting that produces geometric correspondences between fiducials and projector pixels. Then, homography transformation is performed using the correspondences. Instead of using fiducials, Hashimoto et al. [22] employed gray-coded structured lighting and manually refine a reconstructed 3D geometry on which video contents are projected. Notably, a projection surface does not have to correspond to a projector, but convex mirrors are used with four projectors so that five planes

including three walls, a ceiling and a floor can be illuminated simultaneously. From raw images of the projections taken with different shutter speeds of a CCD camera image, gamma correction of overlapping regions of the projections and surface color compensation are achieved simultaneously.

In augmented reality research, one of the earliest video projection applications unconstrained by a tabletop is known as an object-adaptive display, introduced by Raskar et al. [39]. With a handheld device consisting of a calibrated projector-camera pair, the camera captures a fiducial, or *piecode*, similar to a piechart segmented by different colors, in a physical environment so that the label and pose of the fiducial can be determined. Since the projector is calibrated to the camera, the fiducial in the projector image plane is estimated, and video texture can be projected on the physical surface. Lee et al. [28] used a fixed projector and photosensors affixed to an object to reliably locate the object pose for video projection, which is described as marker tracking in Section 2.3. Recently, IllumiRoom [25] demonstrated projection on the peripheral of a television screen (e.g., walls and furniture); unlike most of the CAVE-like systems which treat the environment as a screen on which the users focus, they use projection as a peripheral display that does not attract the users but achieves optical illusion or matches the appearance to the virtual world to enhance the user experience. To acquire the 3D geometry surrounding the television, a Kinect is calibrated to the projector by gray-coded structured lighting, and then the point cloud captured by the Kinect is warped to the projector coordinates for video mapping. The television screen is automatically detected by displaying fiducials on the screen,

which are observed by the Kinect device. Jones et al. claim that the system can be deployed in typical living rooms.

Artists have been adapting projection mapping techniques to art installations, which are often non-interactive and are rather considered as spectacles compared to virtual reality or augmented reality systems. They projected videos onto building facades including landmarks, such as Sagrada Familia [16], Tokyo Station [50] and the Sydney Opera House [38] or any other static objects to change the appearance of their textures or shapes by optical illusion. Such installations often use commercial software, for example, vvvv [51], MadMapper [30] and X-Agora [52], which allows users to define virtual facades and video contents to be mapped on the physical facades by projectors calibrated by manually matching projected pixels to the scene. Projection mapping at Tokyo Station took approximately seven hours for calibration as 46 projectors were deployed [14]. Video projection can be seen in theatre preformances as well; for example, Delirium by Cirque du Soleil with Michel Lemieux and Victor Pilon [11] employed several semi-transparent screens where videos were mapped, and these screens were placed in front of the performers to achieve Pepper's ghost, which can be perceived as a pseudo-holographic effect. As for projection on dynamically moving objects, the video of Bot & Dolly's Box [10] demonstrated projection mapping on a mechanically controlled panel whose motion is predictable. For a target with unrestricted motion, a facial projection mapping achieved by Bell's music video "Chase No Face" [7] was performed by a facial recognition library [29] to track the performer's face by a video camera combined with a Kinect. Furthermore,

projection mapping systems combined with user interaction are emerging in art installations, such as MobiSpray [43], which allows users to optically paint building facades by projection controlled by mobile phones. A phone with a specific application installed is connected to a wireless LAN, and the application turns the phone into a virtual spray can that reads motion sensors and keyboard inputs and sends drawing commands to a rendering server that is connected to projectors.

CHAPTER 3
Static Projection Mapping

As a collaboration with the Theatre at the University of British Columbia (UBC), which uses 6 to 12 projectors for a stage performance, we built a video projection system using a projector and a camera to facilitate projector calibration for static objects with quadrangle surfaces, such as screens on a theatre stage, which is explained in this chapter. The system first scans the geometry of a scene automatically, and the output, a point cloud representation of the scene, is rendered on a monitor. Then, a user selects four points to form a rectangle that corresponds to the physical object surface; here, to simplify the video projection, projected video contents are assumed rectangular, to which homography transformation can be applied for texture mapping. Finally, the 3D coordinates of the selected points are sent to a program that renders pre-warped textures on a projector frame buffer. The scene is assumed static throughout the process.

In Section 3.1, the pipeline of the projection system is explained in detail followed by self camera calibration and homography transformation.

3.1 Methodology

To facilitate video mapping, cameras are used as well as projectors in order to reconstruct the 3D geometry of a scene. Since multiple projectors are often used in art performances, manual calibration of such setups may take a significant amount of time. Therefore, automatic calibration approaches are adopted for facilitation

of projector alignment to the scene. Specifically, structured light is projected and observed to find pixel correspondences among projectors and cameras, and a self-calibration algorithm is applied to find relative poses of the devices, as described in Section 3.1.1. Once the 3D geometry is acquired, a user can select planes where textures will be projected, and a renderer maps the textures to the 3D model for reprojection on the physical scene as explained in Section 3.1.5. The implementation is described along with the framework used, including the specification of configuration files and communication between the software (Section 3.1.6).

Although this section is for explanation of the projection on static objects, projection on dynamically moving objects, which is introduced later in Chapter 4 initially requires 3D scanning by this pipeline.

3.1.1 3D Geometry Acquisition and Camera Calibration

To reconstruct a 3D model of the scene by stereo vision triangulation, first, pixel correspondences between the projector and camera are measured by time-multiplexed gray-coded structured light and phase-shifting sinusoidal patterns [41]. These patterns are most suitable for achieving a dense and high resolution depth map, assuming the scene is static.

Then, for a single projector-camera pair setup, focal lengths, extrinsic parameters and lens distortion coefficients are estimated from the epipolar geometry [53] (Section 3.1.3 and 3.1.4). From the pixel correspondences and estimated parameters, the 3D points are reconstructed by triangulation.

3.1.2 Structured Lighting

Since the target scene is assumed static, time-multiplexing structured light is suitable for 3D scanning as reviewed in Section 2.2. Among the time-multiplexing methods, Gühring's method [19] yields the highest resolution and accuracy, and thus this method is adapted.

First, $\lceil \log_2 m \rceil + \lceil \log_2 n \rceil$ gray-coded patterns are projected, where the resolution of the projector is $m \times n$ pixels, as shown in the top row of Figure 3–1. Then, to recover the pixel correspondences between projector and camera, a codeword is decoded from the first $\lceil \log_2 m \rceil$ gray code at each camera pixel; this codeword uniquely points to a projector column from which the rays are emitted. Similarly, from the last $\lceil \log_2 n \rceil$ patterns, a projector row is obtained. In this manner, the mapping is found between each successfully decoded camera pixel and its corresponding pixel in the projector image plane. In practice, complementary gray-coded patterns are projected for thresholding; thus, a total of $2(\lceil \log_2 m \rceil + \lceil \log_2 n \rceil)$ patterns.

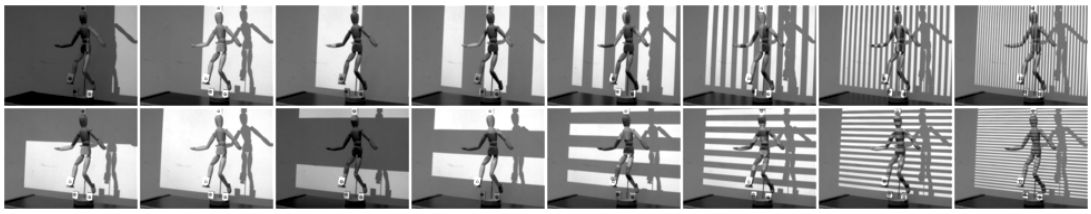


Figure 3–1: Time-multiplexed gray-coded structured light projected on the scene to obtain initial 3D geometry. Top row: first 8 vertical patterns. Bottom row: first 8 horizontal patterns.

Sub-pixel precision can be obtained by the phase-shifted sinusoidal patterns. Each projector pixel (x, y) of the vertical pattern is assigned a phase, $\theta(x) =$

$\frac{2\pi \mod (x,L)}{L}$ where L is the wavelength of the sinusoidal pattern. Theoretically, x is not an integer but can take any real number if the intensity of the observed projection is continuous. The k-th vertical pattern is shifted by the phase $\delta_k = 2\pi k/L$; thus, the observed intensity I at the corresponding camera pixel is formulated as,

$$I = A\sin(\theta(x) + \delta_k) + m \tag{3.1}$$

$$= (\sin\delta_k,\ \cos\delta_k,\ 1)\cdot(A\cos\theta(x),\ A\sin\theta(x),\ m)^T \tag{3.2}$$

where A is the amplitude of the observed sinusoidal pattern and m is the observed bias component. Since δ_k is known, $\theta(x)$ can be solved by applying least squares fitting to a camera pixel with a set of intensities from L images, and consequently, x is acquired with sub-pixel precision. Similarly, horizontal patterns are projected to measure y with sub-pixel precision, as shown in Figure 3–2.

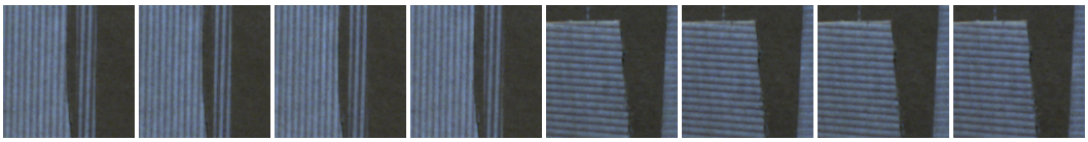


Figure 3–2: Phase-shifted sinusoidal structured light projected on the scene. The first 4 images show vertical patterns, and the last 4 images show horizontal patterns, both at a wavelength of 8 pixels. The images as shown here are cropped to improve visibility of the patterns.

3.1.3 Self Camera Calibration

Given pixel correspondences between a projector and a camera by structured light, self-calibration algorithms can be applied to solve the relative position and pose. Let $\mathbf{m}_c = [u_c\ v_c\ 1]^T$, a pixel of the camera image plane, has the same

codeword as $\mathbf{m}_p = [u_p \; v_p \; 1]^T$, a pixel of the projector image plane, and these two points correspond to the same single physical point $\mathbf{M} = [X \; Y \; Z \; 1]^T$. The geometric relation of the homogenous coordinate points $\mathbf{m}_p$, $\mathbf{m}_c$ and $\mathbf{M}$ is explained by the following proportional expressions,

$$\mathbf{m}_c \sim K_c \begin{bmatrix} \mathbf{I} & \mathbf{0} \end{bmatrix} \mathbf{M} \tag{3.3}$$

$$\mathbf{m}_p \sim K_p \begin{bmatrix} R & t \end{bmatrix} \mathbf{M} \tag{3.4}$$

where $\mathbf{I}$ is a 3×3 identity matrix; $\mathbf{0}$ is a 1×3 zero vector; K_c and K_p are intrinsic camera matrices of camera and projector, respectively; R and t are rotation and translation vectors from the camera to the projector, respectively.

By manipulating the equations above, a fundamental matrix $\mathbf{F}$ and an essential matrix $\mathbf{E}$ are defined as

$$\mathbf{m}_p^T \, \mathbf{F} \, \mathbf{m}_c = 0 \tag{3.5}$$

$$\mathbf{E} = K_p^{-T} \, \mathbf{F} \, K_c^{-1} \tag{3.6}$$

$$= R \, [t] \tag{3.7}$$

where $[t]$ is a matrix form of the outer product defined as

$$[t] = \begin{bmatrix} 0 & -t_3 & t_2 \\ t_3 & 0 & -t_1 \\ -t_2 & t_1 & 0 \end{bmatrix}. \tag{3.8}$$

Since $[t]$ is rank deficient (i.e., rank 2 at most), R and t can be solved up to a scale factor by singular value decomposition (SVD) when $\mathbf{E}$ is known; also $\mathbf{F}$ can be

estimated only from the pixel correspondences. Thus, given pixel correspondences and intrinsic matrices, self-calibration can be achieved. To estimate intrinsic matrices, assuming principal points are known, which is usually the center for a camera and is described in the datasheet for a projector, Bougnoux's method [8] is used. Once the intrinsic and extrinsic matrices are acquired, **M** is solved to reconstruct the 3D geometry.

3.1.4 Radial Fundamental Matrix

Practically, projector and camera lenses have distortion which may not be negligible. To cancel lens distortion mathematically, a radial fundamental matrix, **R**, is computed for the epipolar geometry. Radial distortion is defined as

$$(u',\ v') = \frac{1}{1 + d\sqrt{u^2 + v^2}}(u,\ v) \tag{3.9}$$

where $(u,\ v)$ and $(u',\ v')$ are pixel coordinates without and with distortion, respectively, and d is a distortion parameter. Note that the pixel coordinate is shifted so that its center is on the principal point. Using the lifted coordinate $(u^2 + v^2,\ u,\ v,\ 1)^T$, the distortion equation becomes

$$k \begin{bmatrix} u' \\ v' \\ 1 \end{bmatrix} = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ d & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} u^2 + v^2 \\ u \\ v \\ 1 \end{bmatrix} \tag{3.10}$$

$$= \mathbf{D}\ (u^2 + v^2,\ u,\ v,\ 1)^T \tag{3.11}$$

where k is a non-zero scalar. In the epipolar geometry, radial distortion is applied to both projector and camera; therefore, by denoting the projector and camera distortion matrix by $\mathbf{D}_p$ and $\mathbf{D}_c$ respectively, the matrix $\mathbf{R}$ is defined as

$$\mathbf{R} = \mathbf{D}_p^T \ \mathbf{F} \ \mathbf{D}_c \tag{3.12}$$

where $\mathbf{F}$ is a fundamental matrix. Once the radial fundamental matrix is computed from the pixel correspondence, all the pixels in the epipolar geometry can be undistorted. Also, since the fundamental matrix is available, misdetected pixel pairs can be found by evaluating the following equation

$$\mathbf{u}_p^T \ \mathbf{F} \ \mathbf{u}_c > e \tag{3.13}$$

where $\mathbf{u}$ is an undistorted pixel and e is a threshold that when the left hand side is larger than e, the pixel pair is considered as an outlier.

3.1.5 Texture Mapping

To achieve texture mapping, two methods are proposed: the textured mesh is a rectangle and a designer can map an image to the mesh; or the designer can draw on an image taken by the camera, calibrated with the projector, with a paint tool as if drawing on a 2D image.

With the first method, to pre-warp a texture in the projector image plane, the 3D points of the four corners $\mathbf{X}_i = [X_i \ Y_i \ Z_i \ 1]^T, i = 0 \ldots 3$ of the rectangle in world coordinates are projected onto pixels $\mathbf{u}_i = [u_i \ v_i \ w_i]^T = \mathrm{P}_e \mathbf{X}_i$ in the projector image plane coordinates, where P_e is the projection matrix of the projector. In general, the transformed pixels do not form a rectangle. Thus, the

OpenGL transformation matrix is appropriately set to pre-warp the texture and map to the projector frame buffer using the homography [45] which is solved by

$$\begin{bmatrix} x_i \\ y_i \\ 1 \end{bmatrix} \sim \begin{bmatrix} p_1 & p_2 & p_3 \\ p_4 & p_5 & p_6 \\ p_7 & p_8 & p_9 \end{bmatrix} \begin{bmatrix} u_i \\ v_i \\ 1 \end{bmatrix} \tag{3.14}$$

where $p_1 \dots p_9$ form transformation matrix and the vector $(x_i\ y_i)$ is a corresponding coordinate of the texture.

The second method exploits the texture coordinates of a shader program. Since each 3D point can be located in the camera image coordinate system through structured lighting, the point can be assigned a $(u\ v)$ texture coordinate of the camera image. When virtual painting is superimposed over the camera image, the color of a 3D point will be updated accordingly, and mapped on a physical surface. Although the 3D point cloud can be sparse, Delaunay triangulation [9] generates a mesh for interpolation.

3.1.6 OpenFrameworks and Implementation

OpenFrameworks [35] is a cross-platform and open-source C++ framework developed for building interactive art installations and generating algorithmic visual effects, often referred to as creative coding. It enables developers to code OpenGL 3D rendering, image and video rendering, input device event handling, filesystem I/O and miscellaneous operations concisely since frequently used codes are represented by its application programming interface (API). Furthermore, by including openFrameworks add-ons (ofxAddons), extra API wrappers can be integrated: for example, APIs for OpenCV (a computer vision library), Lapack

(a library for efficient matrix operations), NaturalPoint NatNet (a library for communicating specifically with the Optitrack motion capture system) and the Point Cloud Library. All the software in this thesis is built on openFrameworks to exploit OpenCV, Lapack and its miscellaneous graphics library.

To facilitate developing projection mapping systems, self-calibration software by Yamazaki et al. [53] is used to build ofxActiveScan [34], an open-source wrapper for openFrameworks. Since ofxActiveScan is designed for a projector-camera pair, its information, including captured structured light images, calibration parameters, and a point cloud, is stored in a single data folder to keep the files organized. A data folder includes a calibration config file with the following parameters:

- `proWidth/proHeight/camWidth/camHeight`: Projector screen buffer size and camera image size.
- `grayHigh`: White value for structured light (0-255). It must be carefully adjusted to avoid inter-reflection between adjacent surfaces.
- `devID`: Software device ID of the camera.
- `bufferTime`: Buffer time for structured light capturing in milliseconds for synchronization of the projector and camera.
- `vertical_center`: y value of the principal point of the projector divided by image height (0: top of the image, 1: bottom), which is proportional to the lens shift. This can be calculated from the distances a' and b' in Figure 3–3 as

$$y = \frac{a'}{a' + b'} = \frac{a}{a + b}. \tag{3.15}$$

The values of a' and b' can be found in a user manual of the projector, e.g., $a' = 102$ cm, $b' = -11$ cm thus $y = 1.12$ for the Mitsubishi XD490U. Note that the vertical center is independent of the focal length and distance to the screen.

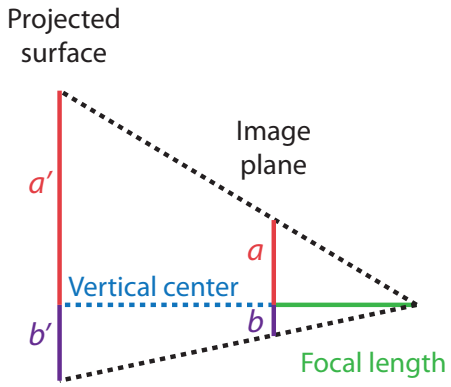


Figure 3–3: A diagram of the vertical center.

Thus, the projector and camera resolution and their principal points must be defined before calibration. The principal point of the projector can be estimated as above, and the principal point of the camera is currently assumed to be the image center (i.e., $y = 0.5$). Using these parameters, an example of the configuration file is shown below.

```
%YAML:1.0
proWidth: 1024
```

```
proHeight: 768
camWidth: 1024
camHeight: 768
grayHigh: 80
devID: 1
bufferTime: 1000
vertical_center: 1.12
```

Once a data folder with a configuration file is prepared, the folder can simply be dragged and dropped to the software window to start calibration. Then the calibration software is made fullscreen on the projector frame buffer, and structured light patterns are projected. From the decoded codewords, the software generates a pixel correspondence map, estimated camera parameters and a 3D point cloud that are written to the data folder.

After calibration and 3D reconstruction, rendering software is launched to visualize a 3D point cloud on a graphical user interface using OpenGL. An arcball interface metaphor is integrated to manipulate the point cloud [44]. The lighting designer then selects four corners to form a rectangle representing a physical surface onto which the designer wishes to project a texture. The texture is specified by dragging and dropping an image file into the point cloud window. The rectangle point coordinates and the texture file path are packed in an Open Sound Control (OSC) message, a protocol to communicate basic types among computers and devices [36]. Homography transformation software receives the OSC message

and maps the texture onto the reprojected quadrangle, which is rendered on the projector frame buffer for video projection.

Figure 3–4: Projector-camera experimental setup.

3.2 Conclusions

In this chapter, 3D reconstruction was achieved by a camera and projector pair by adapting a camera self-calibration method by Yamazaki et al. [53]. Using the proposed method, projection mapping surfaces can be specified in a 3D model, and thus video mapping is expected to be facilitated significantly compared to manual calibration methods, the latter which require a user to map video contents by observing the projected surfaces. Furthermore, the reconstructed 3D geometry is thought to help lighting designers plan stage lighting by rehearsing the projection on a monitor. However, we have yet to develop a stable system to calibrate multiple cameras and projectors, which is a requirement of the Theatre at UBC. Instead of evaluating and improving the usability of the proposed system for the theatre lighting industry, in the following chapters, the system was enhanced

as an interactive system by integrating object tracking methods to explore the application of projection mapping.

A possible enhancement of the system is to use the reconstructed 3D geometry to affect the content of video projection. For example, the geometry can be exploited to automatically extract planar or curved surfaces in a scene, and virtual material or physics can be simulated on the surfaces to produce pseudo lighting or interaction with virtual objects. Implementation of such effects, as well as the process of projection mapping on dynamic scenes, are explained in the following chapter.

CHAPTER 4
Interactive Projection Mapping Using Motion Capture

In this chapter, motion capture is integrated with a projection mapping system described in Chapter 3. With a motion capture system, a moving object can be tracked and projected onto in real-time. Unlike the static projection system, which projects only on planar surfaces, this dynamic projection system is able to project on arbitrary surfaces to which homography transformation cannot be applied: for example, a wooden doll and a facial mask. This chapter not only describes calibration between a projector and motion capture but also introduces a custom marker for object segmentation and tracking.

4.1 Methodology

The kinetic video projection system consists of the following devices: a projector, a color camera, a motion-capture system, and a rigid-body object for the projection target with three or more reflective markers attached. We use a 1024×768 pixel Mitsubishi XD490U projector, a 1032×776 pixel Point Grey Flea2 camera, with output cropped to 1024×768 pixels, and three pre-calibrated NaturalPoint OptiTrack V100:R2 IR cameras, which can robustly estimate retro-reflective marker positions and perform rigid-body fitting for rotation and translation estimation in real-time. First, the projector and camera are self-calibrated using structured light illumination, and 3D geometry is reconstructed as previously described in Chapter 3. Then, reflective markers are detected in

Figure 4–1: Experimental setup. The projector on the right, the three OptiTrack cameras on the left of the image, and the Point Grey camera in front of them are used for the kinetic video projection.

the color camera image by blob extraction for target mesh segmentation and calibration of the projector and motion-capture system. Finally, a mesh is textured and reprojected on the physical object, constantly updated based on the tracked position and pose as obtained from the motion-capture system.

4.1.1 Motion Capture

There are several motion-capture products based on IR active vision, such as Vicon and Optitrack; we use three OptiTrack V100:R2 IR cameras. Such a motion capture system locates spherical retro-reflective markers in a coordinate system by triangulation of the markers which appear as blobs in the IR cameras. Furthermore, three or more locally fixed markers can be used for rigid-body fitting to determine the rotation and translation. Hence, the system can be used for kinetic video projection by attaching reflective markers to a rigid-body object onto which the desired texture is projected. Although the software offers a calibration feature for multiple OptiTrack cameras, projectors and/or color cameras are not supported. Thus, the rigid transformation and scaling factor between the coordinates of the motion capture system and projector must be estimated by users. In general, IR cameras can be calibrated with color cameras using a checkerboard illuminated by an IR light source and applying Zhang's calibration method [56] (Section 2.1), but in our system, since the tracked object is already located in both the motion capture and projector-camera coordinate systems, we propose calibration exploiting their 3D geometry.

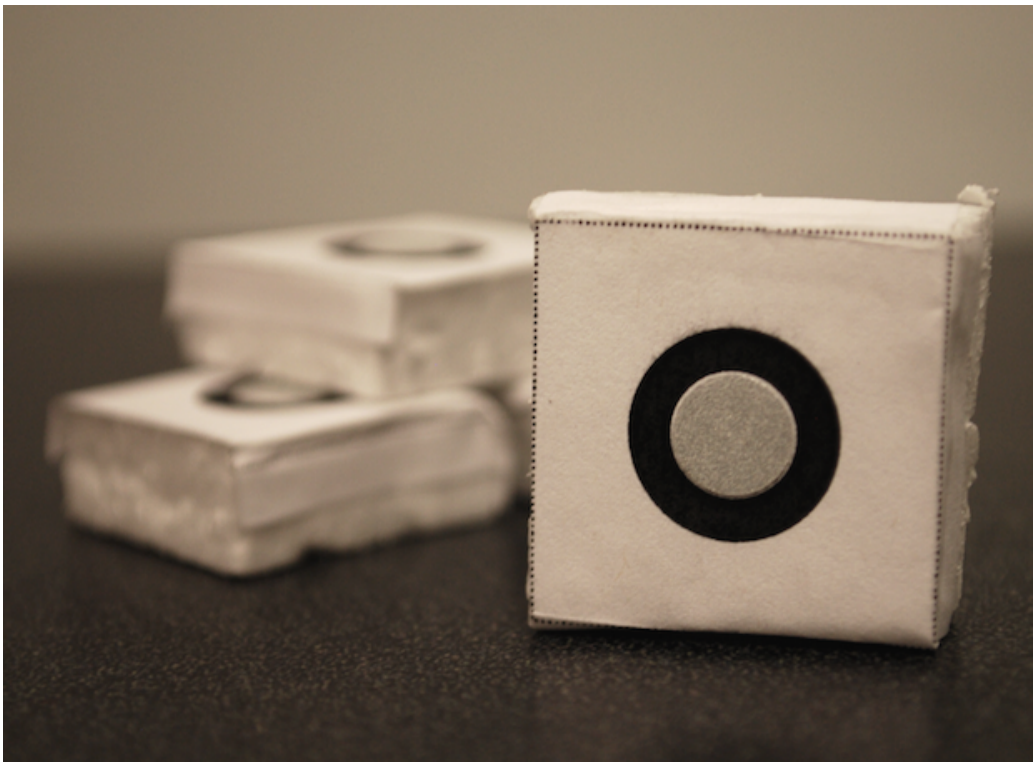
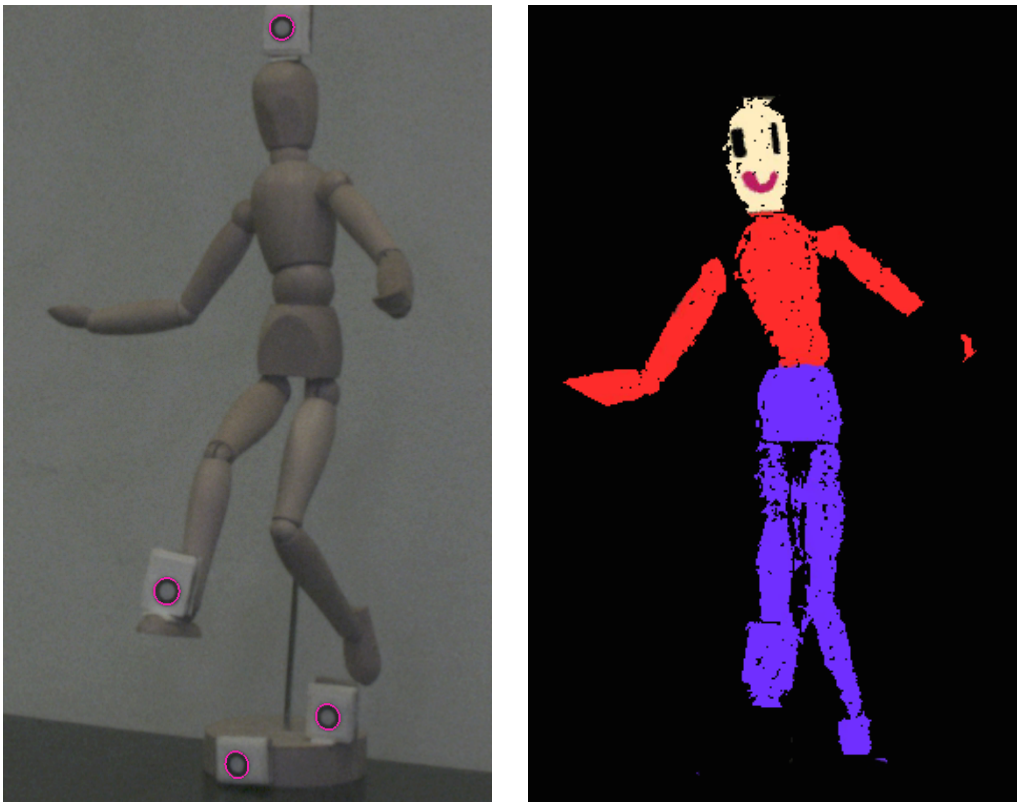


Figure 4–2: Markers for the kinetic video projection system. A retro-reflective sheet cut into a disk, which can be detected by IR active cameras, is surrounded by a black border for blob extraction from a color camera image.



(a) Blob extraction (b) A texture mapped mesh

Figure 4–3: A screenshot of the marker extraction result is shown in (a). The threshold is determined ad-hoc for the best result. In (b), a reconstructed mesh with texture mapped is shown.

4.1.2 Marker Extraction

Instead of using a checkerboard to calibrate the motion capture with the projector-camera pair, custom markers are attached to the projection target so that they can be located in both IR and color camera images. These markers are used for three purposes: mesh segmentation, calibration between the motion-capture system and projector, and object tracking. Each reflective marker consists of an 8 mm diameter retro-reflective disk surrounded by a printed black border (Figure 4–2). The disk center of the markers can be located automatically in the color camera image by blob extraction, using OpenCV, eliminating outliers that do not fit a circle, as shown in Figure 4–3a. Then, the disk center is estimated in the 3D model by averaging the corresponding 3D points within the blob. The 3D coordinates of the disk center are assumed to correspond to one of the markers' coordinates estimated by the motion-capture system.

4.1.3 Mesh Segmentation

After marker extraction, the system automatically segments the projection target object from the background within the point cloud representation. To simplify the segmentation, the 3D points of the target object are assumed to exist in the neighborhood of the marker positions. Specifically, the maximum distance $d_{\max}$ between two arbitrary marker positions is computed, and all the points beyond $\alpha d_{\max}$ from all the markers are filtered out, where α is set to 0.5 assuming that the markers are evenly distributed at distance $0.5h$ from the object center, where h is the longest dimension of the object. However, depending on the object shape, α can be adjusted according to the arrangement of the markers. Finally,

the remaining points are tesselated by the Point Cloud Library [31] to form a mesh with faces.

4.1.4 Calibration of Motion Capture and Projector

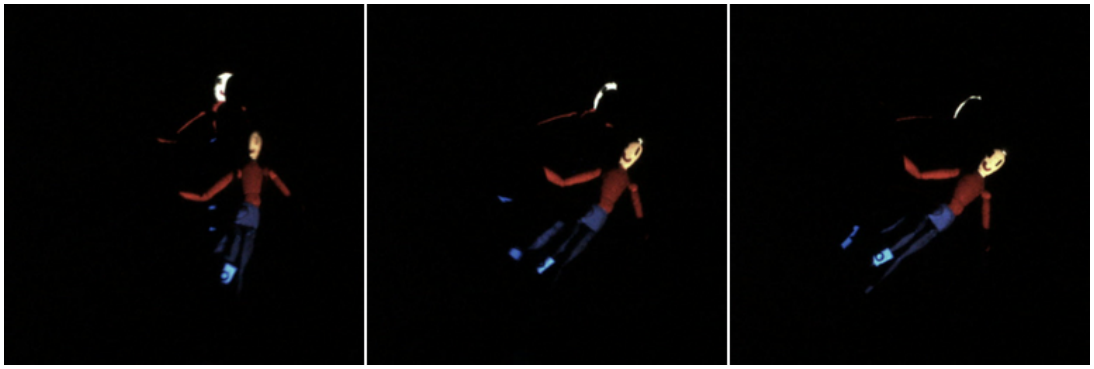
To estimate the rigid transformation and scaling factor between the coordinates of the motion capture system and the projector, the markers extracted from the color image, as described in Section 4.1.2, are matched to the reflective markers tracked by the IR motion capture system. The matching is done by iteratively applying singular value decomposition (SVD) with geometric constraints proposed by Umeyama [49] to find the scaling factor and rigid transformation. The points observed by the motion-capture system and points extracted from the point cloud are represented by V_m and V_p, respectively. $V_{m,p}$ denotes $3 \times n$ matrices with each column vector corresponding to one of the n points. However, since the point correspondence is unknown, the column vectors are unordered, and Umeyama's method cannot be applied. Therefore, a brute-force search method is implemented to find the transformation that minimizes the mean squared error. First, k column vectors in the matrix V_m are randomly chosen to form V'_m. Then, k column vectors in V_p form V'_p, and Umeyama's method is applied to find the transformation matrix M, where $MV'_m = V'_p$. Next, the mean square error between the points MV'_m and V'_p is evaluated. These transform estimation and evaluation steps are iterated for all the permutations for possible V'_p, and the transformation matrix that produces the least mean square error is selected as the transformation between the coordinates. For the working system, k is set to 4. The calibration accuracy is evaluated in Section 4.2.

4.1.5 Video Projection

The rigid-body target is tracked by the motion-capture system, and its rigid transformation matrix M_t at time t in motion-capture space is updated in real-time, and its initial matrix, $M_t|_{t=t_0}$, is the identity matrix. Therefore, the object transformation in the projector coordinates is MM_tM^{-1} using the matrix M estimated above. Finally, using the known pixel correspondences between the projector and color camera, the target object mesh is colored by a texture, drawn on the camera image taken with the initial pose of the object, as shown in Figure 4–3b.

Figure 4–4: Projection of a texture on a dynamic object.

Our implementation uses the OpenGL modelview matrix to represent the transformation for rendering and the projection matrix estimated by self-calibration. To map a texture onto non-planar surfaces, 2D parameterization can



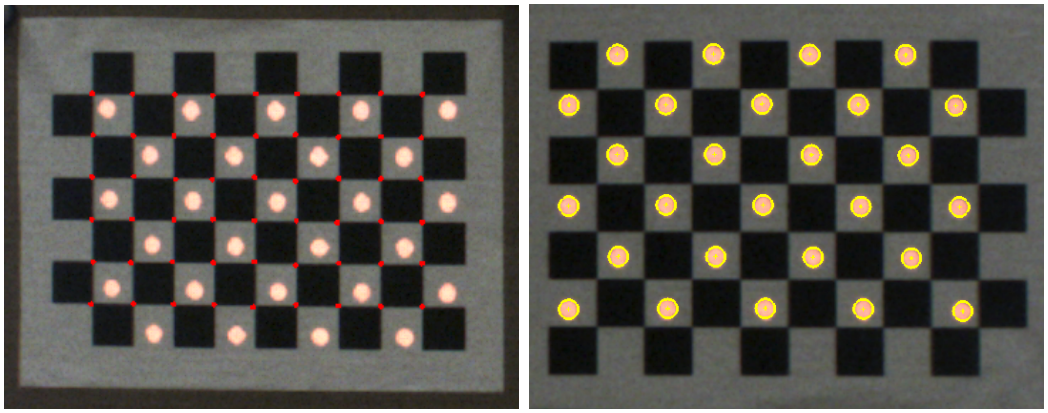
(a) Another example of texture projection. In the frames shown, the wooden figure was moving downward and to the right.

(b) Depth-sensitive kinetic video projection. White lines are rendered at specific depth levels by the OpenGL shader.

Figure 4–5: Illustration of a sequence of kinetic video projection.

be used to define its surface coordinate system. Each point i of the point cloud can be defined a texture coordinate $(u_i, v_i)^T$, which corresponds to the pixel coordinate on the camera image plane, thus $(u_i, v_i)^T = \mathrm{P}_c \mathbf{X}_i$, where P_c is the projection matrix of the camera and $\mathbf{X}_i$ is the 3D coordinate of the point. With the texture coordinates, the camera image overlaid with a texture can be mapped precisely to the point cloud, seen in Figure 4–5a. The overlaying texture can be either predefined using paint software, e.g., Adobe PhotoShop, or updated in real-time as described in Section **??**. If desired, the OpenGL Shader can be programmed to render 3D effects, exploiting the known 3D geometry, as shown in Figure 4–5b.

4.2 Quantitative Results



(a) Checkerboard detection (b) Circle detection

Figure 4–6: For RMSE estimation using homography, first, (a) checkerboard corners are detected from a camera image. (b) Then, a circle pattern is detected from a warped image, and the RMSE is calculated from the circle center displacement.

The projection accuracy on a physical object is evaluated, both statically displaced from an initial position, and while being moved by hand at a certain speed. Although the velocity varies since the target is waved by hand, variance of

the root mean squared error (RMSE) of the projection position is calculated to validate the experimental results. The RMSE of the projection is defined as

$$\text{RMSE} = \sqrt{\frac{1}{N}\sum_{i=0}^{N-1} |\hat{\mathbf{x}}_i - \mathbf{x}_i|^2} \tag{4.1}$$

where N is the number of projected pixels, $\hat{\mathbf{x}}_i$ is a physical 3D point on the target object and $\mathbf{x}$ is a 3D point onto which the corresponding pixel is projected. Practically, we employ a 6×9 checkerboard with square dimension 22×22 mm onto which a 3×9 asymmetric pattern of circles is projected. The circle pattern is prewarped by the homography of the projector and camera in the initial frame and projected onto the printed checkerboard pattern to match the circle centers accurately to the white square centers of the checkerboard. The checkerboard corner points are detected by a color camera, whose lens distortion is corrected beforehand. The observed checkerboard pattern is corrected to be an orthogonal grid by means of the homography [55], which is also used to warp the circle patterns observed in the same image. Consequently, the projection RMSE is simplified to

$$\text{RMSE} = \sqrt{\frac{1}{n}\sum_{i=0}^{n-1} |\hat{\mathbf{u}}_i - \mathbf{u}_i|^2} \tag{4.2}$$

where n is the number of the projected circles; $\mathbf{u}_i$ and $\hat{\mathbf{u}}_i$ denote the 2D position of the center of a circle and a white square on the warped grid, respectively.

The evaluation of accuracy of the kinetic video project system uses a checkerboard on a 320×400 mm grid within the projection volume and a color camera for pattern detection. The X axis corresponds to the horizontal direction in the

projector image plane, and the Z axis corresponds to the optical axis of the projector, projected onto the 2D grid. The checkerboard is initially placed at the grid center $(x, z) = (0\text{ mm}, 1100\text{ mm})$ and manually displaced at intervals of 40 mm to estimate each projection RMSE. The measured RMSEs are converted to pixels in the projector image plane according to the checkerboard depth and plotted in Figure 4–7. The RMSE is 0.89 pixels at the initial position and within 2 pixels and 4 pixels in 71% and 99% of the measured points, respectively.

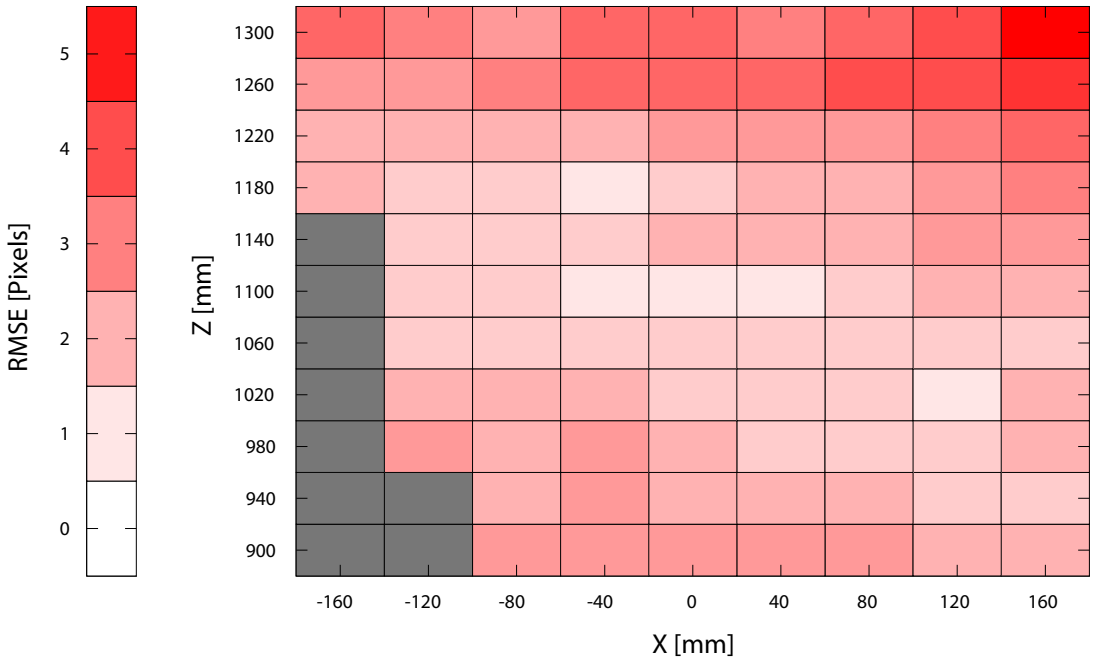


Figure 4–7: Projection RMSE plotted against the XZ plane. The measured RMSEs (in mm) are scaled by the pixel dimension, which varied between 0.43 and 0.63 mm depending on depth. The gray areas indicate RMSE cannot be measured due to the limitation of the projector field of view.

The same checkerboard target was also used to evaluate the response time of the system. The checkerboard was waved by hand along the X axis within a

range of -160 to 160 mm at specific frequencies using a metronome as a reference to estimate the speed. The corresponding RMSE, which we refer to as the dynamic RMSE, was measured when the checkerboard crossed the initial position $(x, z) = (0, 1300 \text{ mm})$. For each speed, dynamic RMSEs were measured 12 times to estimate the mean and standard deviation for validation of accuracy. The results are plotted in Figure 4–8 against the estimated speed of the checkerboard motion. For comparison purposes, the assumed displacement of the checkerboard within the minimum estimated inherent system latency of 38 ms is also plotted. This estimate consists of OptiTrack processing time (4 ms), the OpenGL refresh rate (17 ms at 60 Hz), and the minimum projector latency (also 17 ms at 60 Hz). The measured dynamic RMSEs are approximately 1 mm worse than the displacement, which can be attributed to accumulation of the static RMSE.

4.3 Conclusions

In this chapter, a motion capture system was integrated to a static video projection setup described in Chapter 3. An incremental advance was made to the static projection setup; a projector-camera pair was calibrated by the same method, and the proposed custom tracking marker enabled rigid-body tracking and object segmentation. Unlike the previous system, the acquired 3D geometry was exploited for the OpenGL shader.

The projection accuracy was quantitatively measured using a 1024 $\times$ 768 pixel projector and a manually controlled target object, and determined to exhibit error less than 2 pixels in 71% of a 320 mm $\times$ 400 mm working area. For simplification, we restricted rotation of the target object within the detectable angle of the

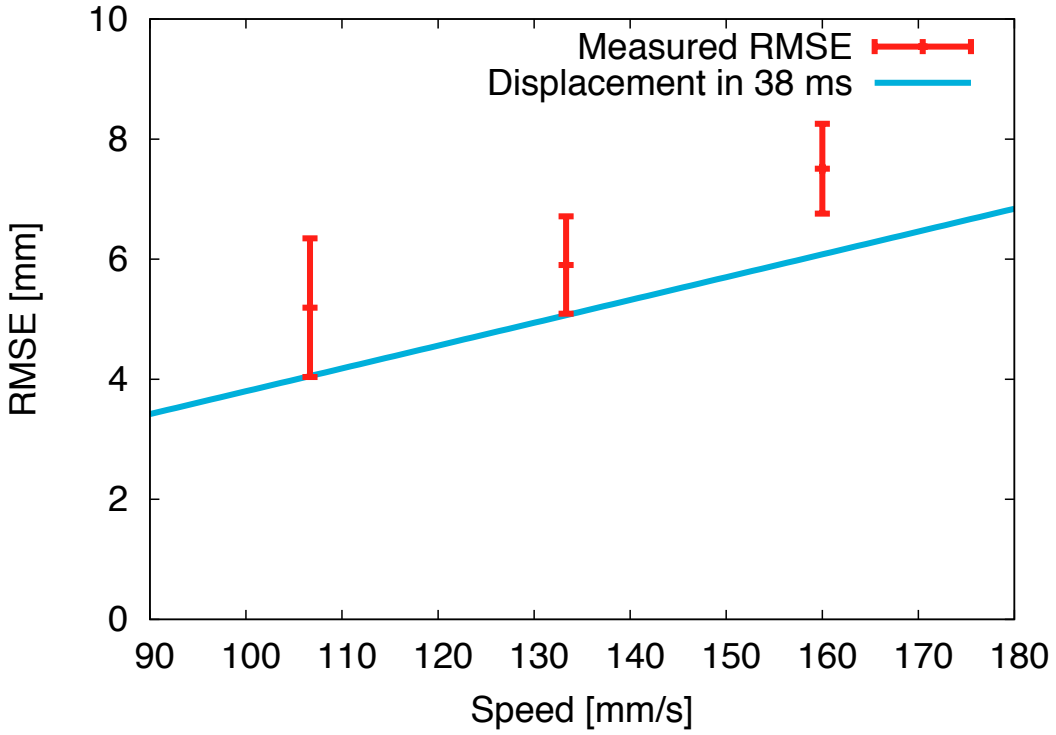


Figure 4–8: Projection RMSE (in mm) on a moving object, as a function of object speed. The RMSE is very close to the amount of object motion during the minimum estimated inherent system latency, and unsurprisingly, is positionally biased behind the object in the direction of its motion.

reflective markers. To further improve the system for robust intrinsic parameter estimation and generalize it to support operation over a greater working volume with potentially more occlusions, multiple projector-camera calibration methods would need to be integrated.

Another experiment with a dynamically moving object confirmed that the system response time is satisfactory given the inherent latency of the equipment. However, provided the estimated latency Δt, the system response can be enhanced by integrating a filter that predicts the object state at $t + \Delta t$ given the state at time t, so the contents are projected with the objective of reducing the delay, which was attempted in the following chapter.

CHAPTER 5
Interactive Projection Mapping Using a Depth Camera

In this chapter, we describe our efforts to develop a facial augmentation system as an entertainment application of the dynamic projection mapping explained in Chapter 4. [1] Not only does the chapter address facial projection, but it also explains virtual drawing on the projected content using a tablet device and a hand tracking sensor. Drawing on a face, in the form of "make-up", represents an important element of entertainment. It offers playful, ludic benefits, e.g., face painting to transform a child into an animal, and even has cultural associations, for example with aboriginal art used in ceremonies. Although such drawing has traditionally relied on the physical application of paint, it is possible to map digital textures to the face using a combination of video projection and face-tracking technologies. Significantly, this approach need not be limited to the application of one-shot face painting, i.e., projection of a static texture, but can be used to project time-varying, dynamic contents on the face, often producing compelling aesthetic results [7, 48]. These examples, however, are designed for art performance. We are unaware of other systems that consider the possibilities of interaction with the projected face. This chapter describes our exploration of the

[1] The contents of this chapter are an edited version of Hieda and Cooperstock, *Digital Facial Augmentation for Interactive Entertainment*, INTETAIN 2015

potential of face projection technology for interactive applications, illustrated by several examples we developed for entertainment purposes.

Since we aim to install the system in exhibitions, we replaced the motion capture system with a Kinect sensor and infrared LED markers, which weigh less for transportation and do not need multiple cameras with tripods. Later, we found that the Kinect face tracking SDK, which performs markerless face tracking to find the face pose and facial features (e.g., jaw lowerer and lip raiser), is useful for facial projection mapping.

5.1 Implementing a Facial Augmentation System

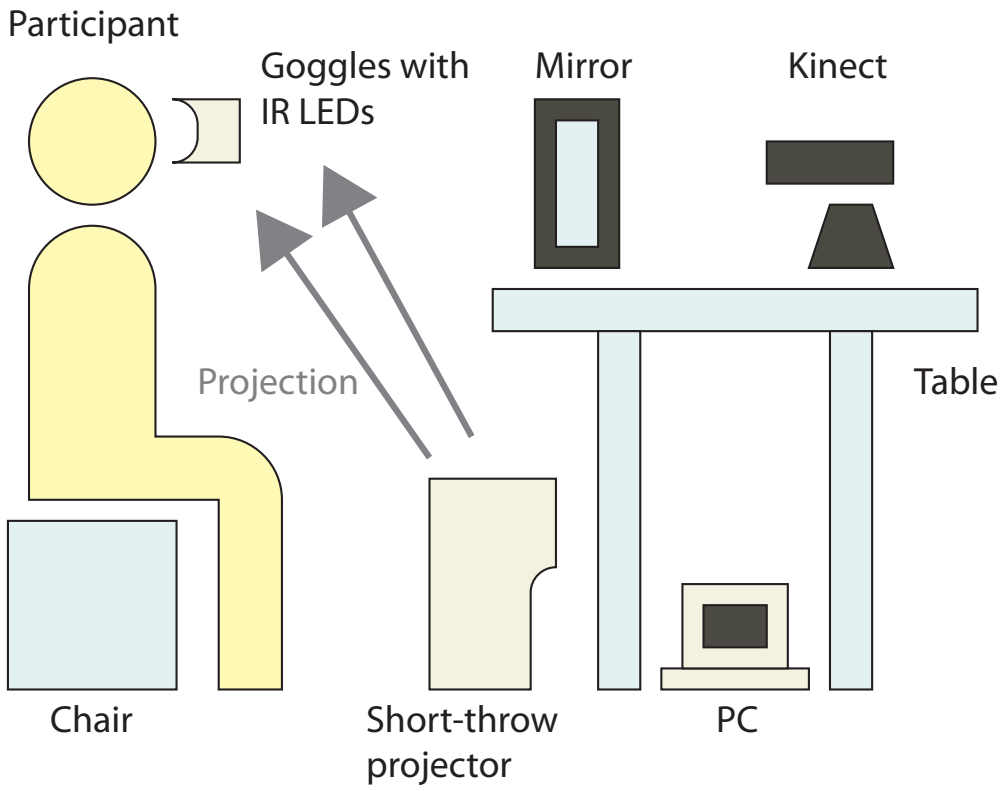


Figure 5–1: Side view of the LED-marker-based prototype.

Figure 5–2: Tablet interface. On the left half of the image, a participant with the goggles is shown. On the right half, stickers are displayed along with pen color selection buttons.

The system uses a depth camera to track the motion of an individual, wearing goggles. Other participants use a digital tablet to draw pictures that are streamed to the video renderer and projected on the physical face by means of a calibrated short-throw projector. Thus, the participants can see their drawings appear on someone's face in real time through projection mapping, without requiring any extra devices. The individual whose face is "drawn upon" can observe the projection through a mirror.

Before projection mapping, the hardware must be calibrated (Section 5.1.1), and a 3D representation of the face, with goggles, must be acquired, with the

depth camera operated in a color image acquisition mode. The data is segmented by eliminating points that are more than 30 cm from the markers, since these are guaranteed to be outside of the confines of the face, and a 3D mesh is formed by Delaunay triangulation [9]. The depth camera is then operated in IR image acquisition mode, thereby tracking the IR LED markers (Section 5.1.2). To compensate for the latency of the system, a prediction filter estimates the position of the user's face (Section 5.1.3). Once the marker coordinates are estimated, the pose of the face is computed by least-squares fitting [4]. The 3D mesh with an updated pose is rendered in the projector framebuffer, with texture as specified by the drawing application (Section 5.1.4).

5.1.1 Calibrating the Hardware

Our LED-marker-based prototype consists of a 1024×768 pixel resolution short-throw projector, goggles with markers, a mirror, a Microsoft Kinect for Xbox 360 depth camera, a tablet device and a PC connected to a WiFi network, as illustrated in Figure 5–1. Four wide-viewing-angle IR LED markers are affixed to the goggles, allowing the participant's face to be tracked by the depth camera. Although the Kinect SDK provides a face-tracking library, the use of goggles is also necessary to protect the individual's eyes from the bright projection light. Otherwise, masking of the texture in the vicinity of the eyes would be necessary.

As a preliminary step, the projector and depth camera are calibrated to obtain the intrinsic and extrinsic camera parameters necessary to align the projection. Since lens distortion of a projector is not negligible, intrinsic calibration

of the projector is performed by ProCamCalib [5], using a 1032 × 776 pixel resolution PointGrey Flea2 camera as a reference to find the focal length and lens distortion coefficients of the projector. Next, the depth camera is calibrated with the projector by gray-code structured lighting [41] to find pixel correspondences. Unlike projector-camera calibration used in the previous systems, the relative pose between the projector and camera is estimated by Levenberg-Marquardt fitting, using the OpenCV camera calibration [1] API since the 3D geometry is already known by the depth camera. Let $\mathbf{M_i}$ denote a 3D point output by the depth camera. The reprojected 2D coordinate $\mathbf{m_i}$ of the point $\mathbf{M_i}$ on the projector image plane is

$$\mathbf{m_i} = \mathbf{K}\ \mathbf{E}\ \mathbf{M_i} \tag{5.1}$$

where the intrinsic matrix $\mathbf{K}$ is known, and the extrinsic matrix has six unknowns for rotation and translation. The reprojection error is

$$\|\mathbf{m_i} - \mathbf{m_i}'\|^2 = \|\mathbf{K}\ \mathbf{E}\ \mathbf{M_i} - \mathbf{m_i}'\|^2 \tag{5.2}$$

where $\mathbf{m_i}'$ is the projector pixel that has the same codeword as a depth camera pixel that corresponds to $\mathbf{M_i}$. Given a set of points, the fitting problem is to minimize the summation

$$\sum_i \|\mathbf{K}\ \mathbf{E}\ \mathbf{M_i} - \mathbf{m_i}'\|^2. \tag{5.3}$$

5.1.2 Depth Camera Marker Tracking

Marker tracking is often achieved by using two or more calibrated cameras, for example, a motion capture system; however, it requires multiple cameras and

(a) Leap Motion hand tracking.

(b) A red sphere rendered close to the fingertip.

(c) Rigid-body simulation.

(d) Fluid simulation for virtual tears and blood.

Figure 5–3: Photos of the improved face projection system. In Figure 5–3a and 5–3b, finger tracking methods are integrated to possibly assist drawing on the own face. In Figure 5–3c and 5–3d, 2D physics engines are applied to render virtual objects.

calibration, which may not be feasible for demonstrations. Instead of a stereo camera setup, we use a depth camera to track a rigid-body object, combining IR and depth images. Since the LED markers of the object appear as blobs in an IR intensity image of the depth camera, these markers can be extracted by the OpenCV blob tracker, which not only returns the 2D blob coordinates (x, y) but also labels the blobs based on the previous coordinates. 2D coordinates of the LEDs can be located in the IR image since the LED emission intensity is greater than IR patterns projected by the depth camera; on the other hand, the depth values around the LEDs cannot be measured. To locate the 3D positions of the LED from the depth image, the depth values of surrounding pixels are averaged, assuming the region surrounding an LED is planar. Our implementation samples the four corners of the bounding box of the blob, and the average depth value is assigned as a the marker depth. Finally, the 2D coordinates and depth values are

converted to 3D world coordinates using the focal length of the depth camera. These 3D coordinates are smoothed by Kalman filtering. Then, the pose of the rigid-body object, i.e., the rotation matrix and translation vector, compared to the initial pose is estimated by least-squares fitting [4] so that the face mesh can be transformed to the current position and orientation. The rotation matrix and translation vector are smoothed by the Kalman filter, and their velocities are used for a prediction filter as explained in the following section.

5.1.3 Prediction Filter

Various sources contribute to the overall system latency. These include the Kinect sensor framerate (33 ms at 30 Hz), the OpenGL refresh rate (17 ms at 60 Hz), the minimum projector latency (also 17 ms at 60 Hz), processing delay, communication time, USB overhead, and operating system scheduling. Fortunately, knowledge of the target object velocity and the system latency can be used to estimate the actual target position using the same prediction step of the Kalman filter, thereby compensating for this latency.

The position estimate is obtained by naively assuming a constant velocity

$$p_{k+\Delta t} = p_k + v_k \ \Delta t, \tag{5.4}$$

and the prediction filter is applied to the Euler angles as well

$$\mathbf{\Theta}_{k+\Delta t} = \mathbf{\Theta}_k + \mathbf{\Omega}_k \ \Delta t, \tag{5.5}$$

where $\mathbf{\Theta}$ and $\mathbf{\Omega}$ are vectors of Euler angles and angular velocities, respectively.

	Speed (pixels/frame)	Line distance (pixels)	Latency (frames)
Without prediction	9.46	42.13	4.45
With prediction	9.62	7.94	-

Table 5–1: Results of latency evaluation. The recording and rendering frame rates are set to 30 frames per second.

To measure the system latency, we used a 4-wheel robot with three IR LED markers. The robot drove at constant velocity, and the marker tracking method in Section 5.1.2 was used to project a line between two markers, aligned perpendicular to the direction of motion. A line is printed between the two markers, and the distance between the projected and printed lines as well as the speed of the robot (i.e., the average speed of the markers) was observed by a color camera. Given a distance d between the two lines and a velocity v, the latency τ is $\tau = \frac{d}{v}$.

The line distance and speed were measured without prediction and with prediction, as shown in Table 5–1. Projected and printed line edges were located by hand. From the results without prediction, the latency is estimated as 4.45 frames (149 ms). With prediction filtering, the geometric difference between projected and printed lines is significantly reduced from 42.13 pixels to 7.94 pixels, which represents a value of 18.8% of the original error.

5.1.4 User Interaction

Participants wear the goggles containing IR LEDs. A small transparent region in front of the eyes allows the participants to view themselves in a mirror. A static picture of the participant with the goggles is captured by the depth camera during

Figure 5–4: Specular reflection shading. Highlights can be seen on the cheek and jaw, and move across the face according to the head orientation.

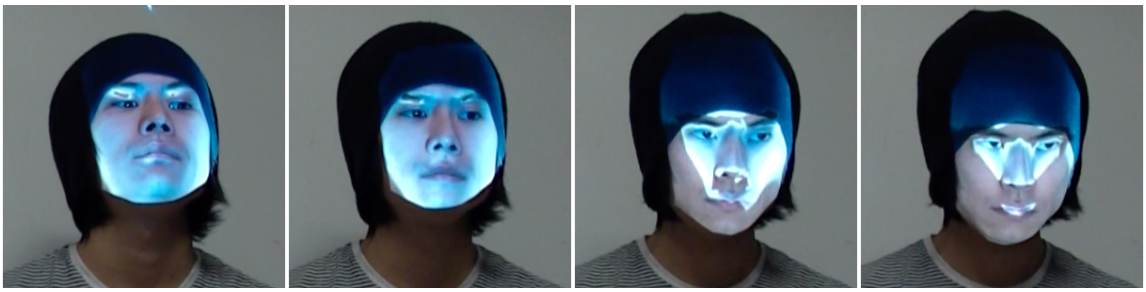


Figure 5–5: Metal reflection shading. Similar to the specular reflection example, surface areas whose normals are parallel to the incident direction of light, i.e., similar to the optical axis of the camera that captured the photos above, are highlighted.

the 3D scanning (Figure 5–2) and displayed on a tablet interface. Using the tablet, other individuals can virtually draw on the participant's face, choosing either static or dynamic pen colors. Additional predefined graphical stickers, such as filled circles and manga-style eyes, can be added to the drawing.

The coordinates of the pen and stickers are sent to a Node.js server running on a main PC through a Websocket. Then, the Node.js server proxies the data by Open Sound Control (OSC) messages [36] to the renderer, which reproduces the drawing on the tablet.

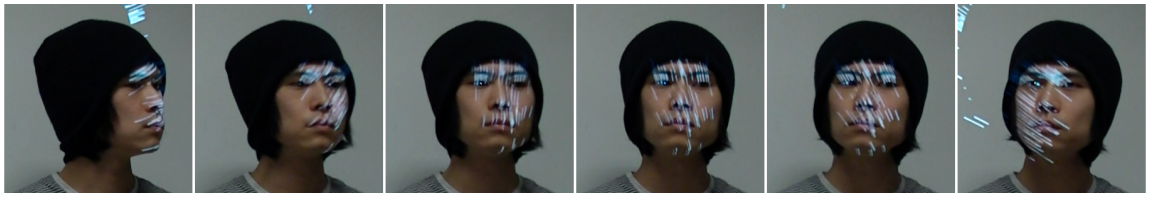


Figure 5–6: Parallax rendering. The projected lines are visible through a mirror, and can be interpreted as parallel line segments extending from the face.

5.2 Enhancements for Effective Interactive Entertainment

The system was tested by over 100 attendees at an international virtual reality venue (Figure 5–7), from which our observations of user behavior led to the formulation of various improvements.

First, drawing on the tablet device can be replaced by aerial gestures or tracing on the face by a physical brush. For this purpose, the enhanced system makes use of a hand-tracking sensor, the Leap Motion device. This frees the participants from having to focus their visual attention on the tablet interface, allowing them to see changes in the projection itself, instantly, while drawing. In addition, the individuals whose faces are augmented can also benefit from this capability, viewing themselves in the mirror while drawing (Fig. 5–3a). Doing so with the tablet interface was challenging at best.

The coordinates of the user's index fingertip are extracted and linearly mapped to the texture coordinate ($0 \leq x < 1024$, $0 \leq y < 768$) of the face projection. To give feedback to the user, a cross-hair cursor is rendered at the corresponding current position on the texture. Although the finger path is continuously connected by a single line, we vary its alpha according to the velocity of the hand. Rapid hand motions, for example, while the user is searching for the

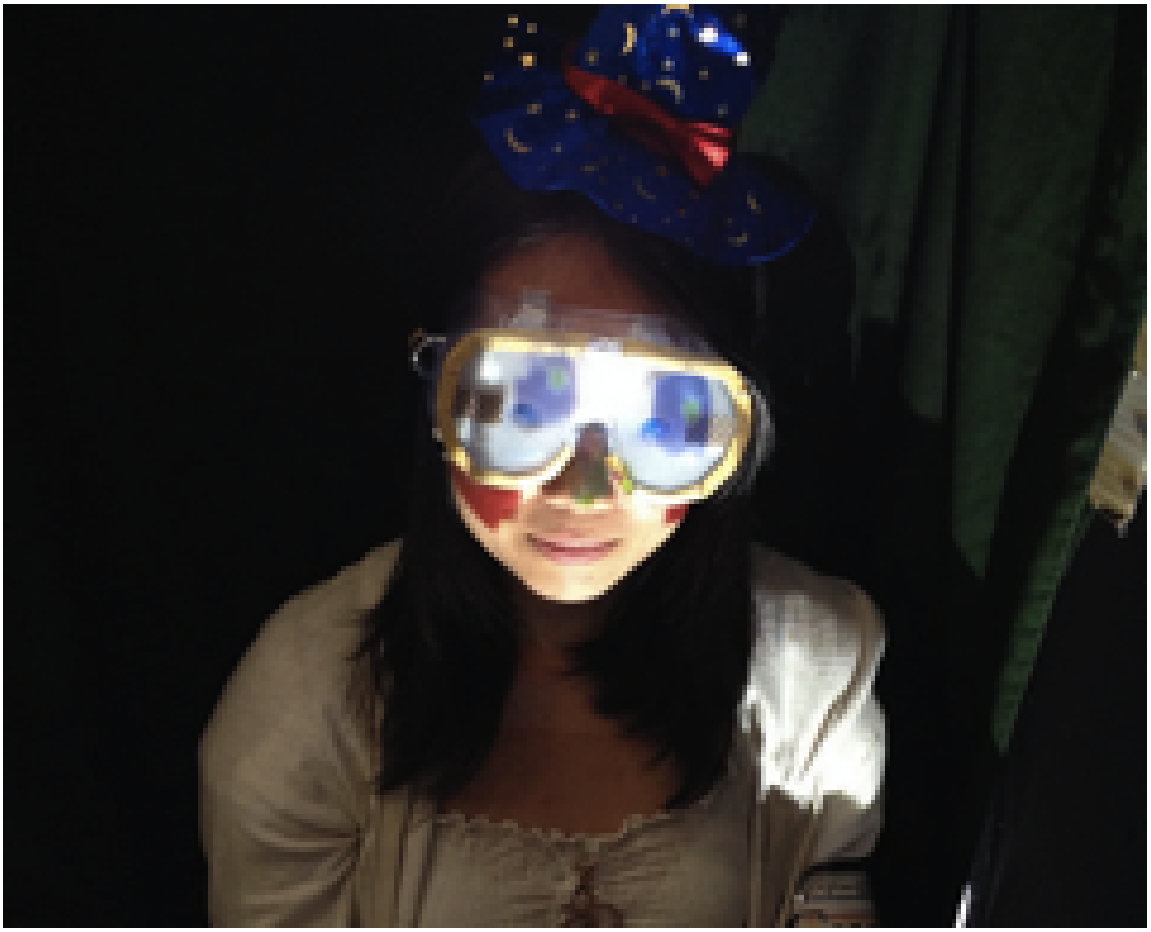


Figure 5–7: A photo taken during the demonstration at the International Collegiate Virtual Reality Contest 2014.

cursor, result in a mostly transparent line, whereas slow drawing gestures result in more opaque lines.

Second, the requirement to wear goggles, which hide the user's eyes, was seen as a significant shortcoming. Fortunately, if the face is not occluded, e.g., by an accessory such as goggles, effective face tracking is possible directly by the Kinect

for Windows SDK [13], without requiring the use of any optical markers. The face-tracking SDK generates a 3D face mesh, whose vertices are sent to our rendering application through Open Sound Control (OSC) messages [36]. The renderer then maps a texture on the mesh (Figure 5–8), setting the OpenGL projection matrix and modelview matrix according to the projector intrinsics and extrinsics, respectively. Finally, the rendered result is projected on the physical face.

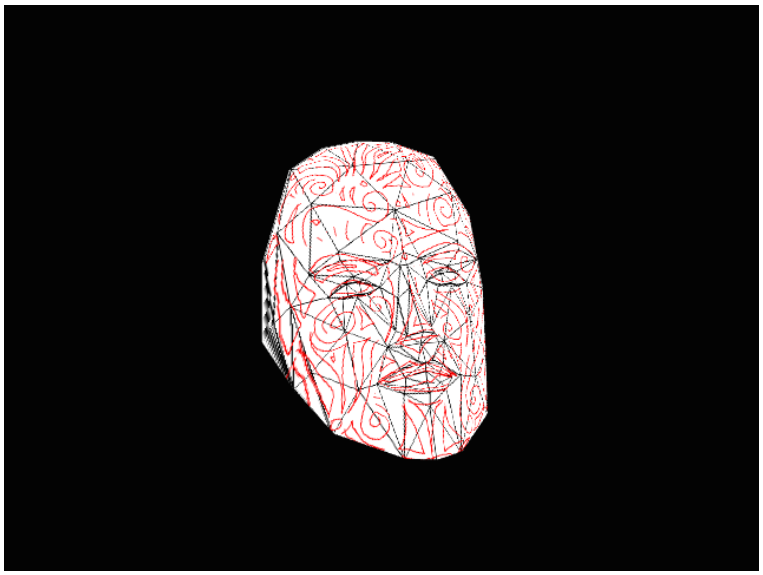


Figure 5–8: An example of a texture-mapped mesh with overlaid wireframe.

Third, 2D physics engines are integrated so that the individual can interact with virtual objects on the face. One example uses a 2D physics engine [2], and a circle rigid-body polygon is generated every two frames around the forehead (Figure 5–3c). On the edge of the jaw, a virtual edge is defined which bounces polygons. The physics is simulated on a 2D plane that is mapped to the face mesh. Although the 2D plane is independent of the world coordinate system of the depth

camera, the gravitational force of the physics engine is rotated in real time to correspond to the world. That is to say, when a head is rotated θ degrees about z axis, which is the axis the nose is pointing, the gravity is rotated $-\theta$ degrees. Thus, the participant can tilt the head to drop the polygons from the jaw. Note that two black circles are overlaid on the eye positions of the final texture to avoid projection on eyes.

Another example, a GPU-based 2D fluid simulator [3] is integrated. In Figure 5–3d, blue and red fluid is generated on the eyes and forehead to map virtual tears and blood, respectively. As in the previous example, the gravitational force is mapped to the world coordinate system. In this example, a virtual obstacle is defined on the nose instead of the jaw, so that fluid can branch at the nose.

Fourth, a primitive example of parallax rendering is tested, by defining a virtual line segment from the face vertices towards the facing direction. As shown in Figure 5–6, the lines, which are projected on the face, rotate to follow the head. From a viewpoint close to the projector, the lines can be seen with a parallax effect. By suitable placement of a mirror, the participant can also enjoy the effect, and can control the orientation of the line by rotating their head. Through an optical illusion, the lines can be thought as virtual line segments fixed to the face, and with appropriate curvature, a virtual "hairy face" can be simulated.

Fifth, relighting is demonstrated using Unity3D built-in shaders. Since the face geometry is known, a virtual light source and virtual face material can be defined to simulate the reflection. In one example, the projected face texture

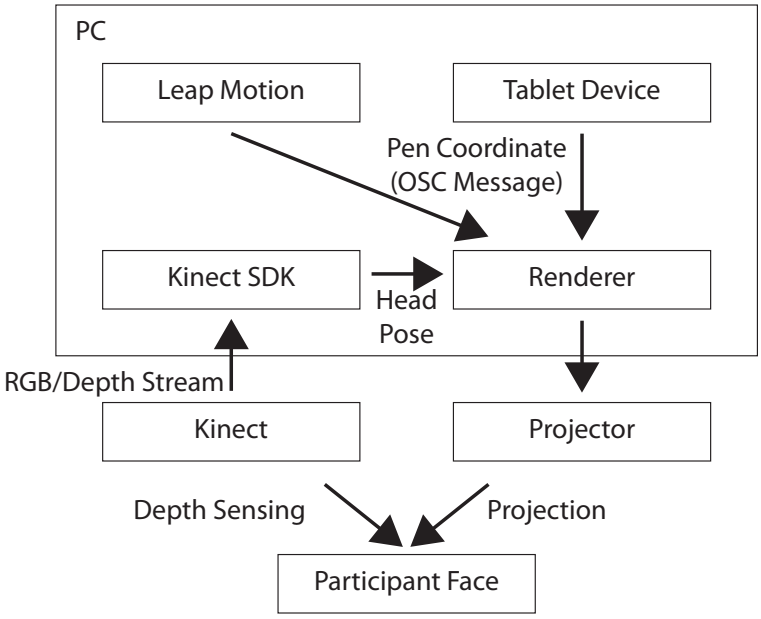


Figure 5–9: A flowchart of the system.

simulates specular reflection highlights on the cheek and jaw (Figures 5–4 and 5–5). This is achieved by placing a virtual, directional light source close to the sensor. Another example shows metallic reflection by physically based shading.

Finally, a simple fingertip detection algorithm is implemented. The algorithm is a simplified version of the method by Harrison et al. [20]. First, the skeletal joint of the head tracked by the Kinect for Windows SDK is projected on a depth map, and a 200×200 pixel region of interest surrounds the projected position (Figure 5–10a). A Sobel map is calculated along the x-axis (Figure 5–10b), as appropriate to identify a vertically oriented finger. Next, the algorithm searches along the x-axis for a convex structure of sufficient size, identified as an array of 30 pixels that transition from negative to zero to positive values. If such a convex structure continues along the y-axis, i.e., corresponding to the edges of the finger, the pixels

(a) A depth map of a face and a hand. (b) A Sobel map. (c) A detected fingertip.

Figure 5–10: Fingertip detection.

found are labeled as a finger, and the top of the continuous structure is extracted as a fingertip (Figure 5–10c). In Figure 5–3b, a detected fingertip is rendered as a red sphere, and projected on the face.

Although such fingertip detection enables a natural interface for facial augmentation, the detection accuracy is limited by the depth resolution of the depth camera. Using commodity hardware, the difference between fingers and the face surface is often not detected. In such cases, both "touch detection" as well as finger detection itself, are not sufficiently robust for this approach to projected face drawing. Furthermore, given the convex structure of the nose, it is sometimes incorrectly detected as a fingertip. More problematically, fingers placed on the face can disrupt the face-tracking algorithm. With the current setup, it is therefore helpful to ensure that the head is stable during interactions when the face is partially occluded.

A flowchart of the tracking, projection and user interaction is shown in Figure 5–9. The proposed system was presented at Laval Virtual 2015, a public exhibition

of virtual reality and augmented reality technologies, similarly to the LED-marker-based prototype (Figure 5–11). Among the proposed effects, fluid simulation was combined with Leap Motion hand tracking. Most of the participants recognized the correspondence between finger motion and fluid simulation on the face.

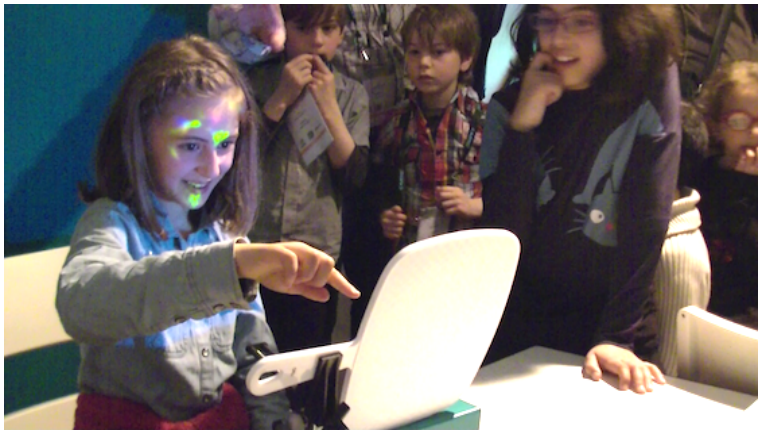


Figure 5–11: A photo taken during the demonstration at Laval Virtual 2015.

5.3 Conclusions

We presented an interactive projection system that maps drawing and virtual objects on an individual's face. Given user feedback on the LED-marker-based prototype of facial projection mapping with markers, we designed a markerless face projection system. With the second prototype, participants do not need to wear devices in order to experience the system. The initial system required use of a pen display device for drawing on a face. We then integrated a hand-tracking sensor to improve the interaction experience for the individuals whose faces are "drawn upon", allowing them to draw on their own faces while viewing the results in a mirror. For this purpose, we implemented a simple fingertip-detection

algorithm; however, the limitations of commodity depth sensors and our simplified assumptions of finger geometry limit the robustness of this approach.

To illustrate other application examples, we employed 2D physics engines for rigid-body and fluid simulation. These allow the individuals to tilt their heads to control virtual objects on the face. We also implemented re-lighting shaders to simulate virtual skin materials such that facial highlights change according to head position and orientation. Finally, we described a primitive parallax rendering example to demonstrate the possibility of integrating 3D rendering. More sophisticated augmented reality applications can be built with 3D bullet physics simulation exploiting a face 3D geometry to interact with virtual objects as well as face re-lighting and parallax effects from the previous examples.

Although several effects are explored in this chapter, human perception to such effects has not been studied yet. As an entertainment system, the system can be extended for multiple participant such as drawing on another participant by facing each other, or using a bigger mirror.

CHAPTER 6
Conclusions

In this thesis, video projection techniques for static and moving objects are presented. The first prototype was a toolkit to facilitate projection mapping on a static scene, by means of 3D geometry acquisition. 3D reconstruction is performed by gray-code structured lighting and self-camera calibration, and thus, manual intervention is not required. The reconstructed 3D point cloud can be manipulated by users to plan video projection on the scene. By selecting its surface corners, the texture can be mapped automatically to the corresponding physical scene by projection. This toolkit maps image or video textures based on stereo correspondences but does not perform surface analysis of the 3D geometry nor dynamic projection mapping. However, since 3D position and orientation of the projector and camera can be estimated through self-calibration, real-time projection on an arbitrary object is possible as long as its pose is tracked. In the second system, which maps videos on a dynamic object, we integrated a motion capture system, which detects retro-reflective markers for rigid-body object tracking. Notably, a custom marker was designed to facilitate mapping between a camera image and motion capture data. In terms of projected video contents, not only static images but also geometry-aware shaders were implemented so that moving the projected objects enables interaction with the virtual world.

Furthermore, the Unity 3D game engine was integrated to facilitate physically based rendering.

We found that dynamic projection mapping is useful for another application, an interactive art installation. Especially, digital facial augmentation has a playful application such as virtual face painting for kids and can be used for virtual makeup without physical painting. To achieve facial projection mapping with our initial prototype, goggles with transparent peepholes were prepared as a projection target to be worn by a participant, and the projection is visible to the participant through a mirror. The goggle tracking was then replaced with bare face tracking provided by Microsoft Kinect sensor. Interaction with projected content was accomplished by using a tablet device running a paint application, which sends pen information to the rendering server to update the projection. Similarly, hand tracking was implemented for aerial virtual drawing to support drawing on one's own face.

6.1 Future Work

As future work, the following techniques could be adopted to improve our projection mapping system. First, for static video projection, a multiple projector-camera system should be supported for projection mapping on a larger scene: for example, a building facade or a theatre stage, in which we originally intended to install the system. A multiple camera calibration method has been proposed by Svoboda et al. [46] and a multiple projector-camera setup is expected to be calibrated by this method. In such a system, some regions are projected by two or more projectors, which requires gamma correction for seamless blending [22].

Second, we ignored the spectral reflectance of a projected surface and no color compensation has been done. Since the projection targets we used do not have uniform spectral reflectance, projected video contents appear in different colors on the surface without color compensation. The surface spectral reflectance can be measured from camera images, and the projection color can be photometrically adapted to the surface [17]. However, the method by Fujii et al. assumes a coaxial projector-camera system using a beam splitter, which differs from our setup. Theoretically, a depth camera perspective can be virtually shifted to simulate a coaxial setup using its depth and color images if the camera is calibrated with a projector. Nonetheless, the effect of the commodity depth camera latency (30 frames per second) has to be taken into account since their setup uses a color camera with a frame rate of at most 60 frames per second.

Third, since a participant's face can move in a large volume, the focus of a projector has to be adjusted accordingly not to lose fine details of the projection. Without lens adjustment, a projector defocus model has been studied by Zhang and Nayar [54]. Optical blur of the projection can be compensated by image convolution of a defocus kernel depending on the surface distance from the projector. Since a head position can be measured by a depth camera, the defocus kernel can be computed in real-time.

The facial projection mapping systems described in this thesis were exhibited at International Collegiate Virtual Reality Contest 2014 and Laval Virtual ReVolution 2015, where participants could perceive the mapped video contents on their face. Nonetheless, we have only showed basic drawing tools, such as a pen

and stickers. Our next step is to build applications of the system; for example, a larger mirror can be placed to imitate a dresser or a bathroom vanity with projection mapping makeup. We continue to explore additional possibilities for the system, which are not limited to entertainment but can be useful for such applications as cosmetic advertisement and health monitoring.

References

[1] Camera Calibration and 3D Reconstruction `http://docs.opencv.org/modules/calib3d/doc/camera_calibration_and_3d_reconstruction.html`.

[2] Box2D — A 2D Physics Engine for Games `box2d.org`.

[3] GPU Gems - Chapter 38. Fast Fluid Dynamics Simulation on the GPU `http://http.developer.nvidia.com/GPUGems/gpugems_ch38.html`.

[4] K.S. Arun, T.S. Huang, and S.D. Blostein. Least-squares fitting of two 3-d point sets. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, PAMI-9(5):698–700, Sept 1987.

[5] S. Audet and M. Okutomi. A user-friendly method to geometrically calibrate projector-camera systems. In *Computer Vision and Pattern Recognition Workshops, 2009. CVPR Workshops 2009. IEEE Computer Society Conference on*, pages 47–54, June 2009.

[6] Samuel Audet, Masatoshi Okutomi, and Masayuki Tanaka. Augmenting moving planar surfaces robustly with video projection and direct image alignment. *Virtual Reality*, 17(2):157–168, 2013.

[7] Chase No Face / BELL. `http://vimeo.com/26649425`.

[8] S. Bougnoux. From projective to euclidean space under any practical situation, a criticism of self-calibration. In *Computer Vision, 1998. Sixth International Conference on*, pages 790–796, 1998.

[9] Paul Bourke. Efficient triangulation algorithm suitable for terrain modelling. In *Proc. Pan Pacific Computer Conf*, 1989.

[10] Box by Bot and Dolly. `https://www.youtube.com/watch?v=lX6JcybgDFo`.

[11] Delirium by Cirque du Soleil. `http://www.cirquedusoleil.com/en/press/news/2005/news102.aspx`.

[12] Human Face Video Mapping by Oskar and Gaspar. http://vimeo.com/39697056.

[13] Qin Cai, David Gallup, Cha Zhang, and Zhengyou Zhang. 3d deformable face tracking with a commodity depth camera. In Kostas Daniilidis, Petros Maragos, and Nikos Paragios, editors, *Computer Vision ECCV 2010*, volume 6313 of *Lecture Notes in Computer Science*, pages 229–242. Springer Berlin Heidelberg, 2010.

[14] NHK (Japan Broadcasting Corporation). Science Zero, broadcasted at Nov 25, 2012.

[15] Carolina Cruz-Neira, Daniel J. Sandin, and Thomas A. DeFanti. Surround-screen projection-based virtual reality: The design and implementation of the cave. In *Proceedings of the 20th Annual Conference on Computer Graphics and Interactive Techniques*, SIGGRAPH '93, pages 135–142, New York, NY, USA, 1993. ACM.

[16] Moment Factory Sagrada Familia. https://www.youtube.com/watch?v=nX9RVDvOCrM.

[17] K. Fujii, M.D. Grossberg, and S.K. Nayar. A projector-camera system with real-time photometric adaptation for dynamic environments. In *Computer Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Computer Society Conference on*, volume 1, pages 814–821 vol. 1, June 2005.

[18] I. Garcia-Dorado and J. Cooperstock. Fully automatic multi-projector calibration with an uncalibrated camera. In *Computer Vision and Pattern Recognition Workshops (CVPRW), 2011 IEEE Computer Society Conference on*, pages 29–36, June 2011.

[19] Jens Guehring. Dense 3d surface acquisition by structured light using off-the-shelf components. *Proc. SPIE*, 4309:220–231, 2000.

[20] Chris Harrison, Hrvoje Benko, and Andrew D. Wilson. Omnitouch: Wearable multitouch interaction everywhere. In *Proceedings of the 24th Annual ACM Symposium on User Interface Software and Technology*, UIST '11, pages 441–450, New York, NY, USA, 2011. ACM.

[21] Richard Hartley and Andrew Zisserman. *Multiple view geometry in computer vision*. Cambridge university press, 2003.

[22] Naoki Hashimoto, Kenji Honda, Makoto Sato, and Mie Sato. Making an immersive projection environment with our living room. In *Proceedings of the 8th International Conference on Virtual Reality Continuum and Its Applications in Industry*, VRCAI '09, pages 83–87, New York, NY, USA, 2009. ACM.

[23] Eli Horn and Nahum Kiryati. Toward optimal structured light patterns. *Image and Vision Computing*, 17(2):87 – 97, 1999.

[24] OpenKinect: Imaging Information. `http://openkinect.org/wiki/Imaging_Information`.

[25] Brett R. Jones, Hrvoje Benko, Eyal Ofek, and Andrew D. Wilson. Illumiroom: Peripheral projected illusions for interactive experiences. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '13, pages 869–878, New York, NY, USA, 2013. ACM.

[26] Martin Kefer and Wilfried Kubinger. Evaluation of kinect depth sensor for use in mobile robotics.

[27] Johnny Lee, Scott Hudson, and Pau Dietz. Hybrid infrared and visible light projection for location tracking. In *Proceedings of the 20th Annual ACM Symposium on User Interface Software and Technology*, UIST '07, pages 57–60, New York, NY, USA, 2007. ACM.

[28] Johnny C. Lee, Paul H. Dietz, Dan Maynes-Aminzade, Ramesh Raskar, and Scott E. Hudson. Automatic projector calibration with embedded light sensors. In *Proceedings of the 17th Annual ACM Symposium on User Interface Software and Technology*, UIST '04, pages 123–126, New York, NY, USA, 2004. ACM.

[29] P. Lucey, J.F. Cohn, T. Kanade, J. Saragih, Z. Ambadar, and I. Matthews. The extended cohn-kanade dataset (ck+): A complete dataset for action unit and emotion-specified expression. In *Computer Vision and Pattern Recognition Workshops (CVPRW), 2010 IEEE Computer Society Conference on*, pages 94–101, June 2010.

[30] MadMapper. `http://www.madmapper.com`.

[31] Zoltan Csaba Marton, Radu Bogdan Rusu, and Michael Beetz. On fast surface reconstruction methods for large and noisy point clouds. In *Robotics*

and Automation, 2009. ICRA '09. IEEE International Conference on, pages 3218–3223, may 2009.

[32] Raymond A Morano, Cengizhan Ozturk, Robert Conn, Stephen Dubin, Stanley Zietz, and Jonathan Nissanov. Structured light using pseudorandom codes. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 20(3):322–327, 1998.

[33] Richard A. Newcombe, Shahram Izadi, Otmar Hilliges, David Molyneaux, David Kim, Andrew J. Davison, Pushmeet Kohi, Jamie Shotton, Steve Hodges, and Andrew Fitzgibbon. Kinectfusion: Real-time dense surface mapping and tracking. In *Mixed and Augmented Reality (ISMAR), 2011 10th IEEE International Symposium on*, pages 127–136, Oct 2011.

[34] ofxActiveScan repository. `https://github.com/micuat/ofxActiveScan`.

[35] openFrameworks. `http://openframeworks.cc`.

[36] opensoundcontrol.org. `http://opensoundcontrol.org`.

[37] J.L Posdamer and M.D Altschuler. Surface measurement by space-encoded projected beam systems. *Computer Graphics and Image Processing*, 18(1):1 – 17, 1982.

[38] Sydney Opera House Facade Projection. `http://vimeo.com/45835867`.

[39] Ramesh Raskar, Jeroen van Baar, Paul Beardsley, Thomas Willwacher, Srinivas Rao, and Clifton Forlines. ilamps: Geometrically aware and self-configuring projectors. In *ACM SIGGRAPH 2006 Courses*, SIGGRAPH '06, New York, NY, USA, 2006. ACM.

[40] J. Salvi, J. Batlle, and E. Mouaddib. A robust-coded pattern projection for dynamic 3d scene measurement. *Pattern Recogn. Lett.*, 19(11):1055–1065, September 1998.

[41] Joaquim Salvi, Jordi Pags, and Joan Batlle. Pattern codification strategies in structured light systems. *Pattern Recognition*, 37(4):827 – 849, 2004. Agent Based Computer Vision.

[42] Tatsuo Sato. Multispectral pattern projection range finder. *Proc. SPIE*, 3640:28–37, 1999.

[43] Jürgen Scheible and Timo Ojala. Mobispray: Mobile phone as virtual spray can for painting big anytime anywhere on anything. In *ACM SIGGRAPH 2009 Art Gallery*, SIGGRAPH '09, pages 5:1–5:10, New York, NY, USA, 2009. ACM.

[44] Ken Shoemake. Arcball: a user interface for specifying three-dimensional orientation using a mouse. In *Graphics Interface*, volume 92, pages 151–156, 1992.

[45] R. Sukthankar, R.G. Stockton, and M.D. Mullin. Smarter presentations: exploiting homography in camera-projector systems. In *Computer Vision, 2001. ICCV 2001. Proceedings. Eighth IEEE International Conference on*, volume 1, pages 247–253 vol.1, 2001.

[46] Tomáš Svoboda, Daniel Martinec, and Tomáš Pajdla. A convenient multi-camera self-calibration for virtual environments. *Presence: Teleoperators and Virtual Environments*, 14(4):407–422, 2013/12/04 2005.

[47] N 520437 E 041900 [the hague city hall]. `http://www.pablovalbuena.com/selectedwork/n-520437-e-041900/`.

[48] Omote: Real time Face Tracking and Projection Mapping. `http://vimeo.com/103425574`.

[49] S. Umeyama. Least-squares estimation of transformation parameters between two point patterns. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 13(4):376–380, 1991.

[50] Tokyo Station Vision. `https://www.youtube.com/watch?v=MQ1djdXuo7g`.

[51] vvvv. `http://vvvv.org`.

[52] X-Agora. `http://xagora.ca`.

[53] S. Yamazaki, M. Mochimaru, and T. Kanade. Simultaneous self-calibration of a projector and a camera using structured light. In *Computer Vision and Pattern Recognition Workshops (CVPRW), 2011 IEEE Computer Society Conference on*, pages 60–67, 2011.

[54] Li Zhang and Shree Nayar. Projection defocus analysis for scene capture and image display. *ACM Trans. Graph.*, 25(3):907–915, July 2006.

[55] Zhengyou Zhang. A flexible new technique for camera calibration. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 22(11):1330–1334, Nov 2000.

[56] Zhengyou Zhang, Rachid Deriche, Olivier Faugeras, and Quang-Tuan Luong. A robust technique for matching two uncalibrated images through the recovery of the unknown epipolar geometry. *Artificial Intelligence*, 78(1–2):87–119, 1995. Special Volume on Computer Vision.