Real Time Perspective Correction of Projector Image Using Structure Sensor Depth Camera

Duc Nguyen

Abstract

I investigate a new approach for determining the pre-warp homography required to correct a projector image for perspective when projecting on a single flat surface. By using a depth camera placed at a fixed transform near the projector, the normal of the surface can be predicted in real time. By updating a homography on the image to be projected based on the projector intrinsics and the surface normal, the image can be correctly pre-warped in real time. This allows for applications where the surface and the projector/camera pair can be moved while the projection remains rectangular and of the correct aspect ratio when viewed from a position normal to the image.

1 Introduction and Motivation

In creative coding, augmented reality, and other entertainment applications, projection onto a non-level, moving surface is of interest for creating exciting visualization experiences. When the object surface (the surface on which the image will be projected) orientation with respect to the projector is unknown, image processing techniques are required to rectify the image for perspective. In order to retrieve the orientation of the object surface, past approaches have displayed a calibration image and then retrieved the homography between the calibration image and a perspective-corrected target image. With recent advances in depth camera technology, including consumer products such as the XBox Kinect, the Structure Sensor, and the DepthSense, the orientation of the object surface as well as a correction homography can be computed in real time.

This project presents a procedure for collecting the depth camera image, preprocessing the image, computing the normal of the object surface, and applying the correct pre-warp homography. Although this project focuses only on projection onto a single flat surface, in the future, it may be expanded to account for arbitrary surfaces as well. For experimentation, the Structure Sensor depth camera was used alongside a Sony MP-CL1 projector.

2 Related Work

Existing approaches [1,2,3,4] on this topic can be classified into two groups: parametric approaches and non-parametric approaches. All of them involve displaying some sort of calibration image in order to solve for a desired homography. In the parametric approach, the image is mapped onto the surface in a texture-mapping like fashion. Thus, the image will look correct when viewed from a vantage point normal to the surface and level to the image. In the non-parametric approach, the displayed image is based on the viewer's point position. Due to contours on the object surface, if the viewer walks away from the predetermined viewing position, the image will appear distorted.

This project focuses on the parametric approach, and assumes that the viewer will always be close to normal to the projector screen.

3 Equipment Used

The Structure Sensor is a depth camera manufactured by Occipital that is capable of capturing 30-60 frames per second. It has a 16:9 aspect ratio with 45 degree vertical FOV. It is precise to 30mm at 3m. By using the OpenNI library, depth images can be extracted from the device for which each pixel is an integer value representing the millimeter distance from the image plane.

The projector used is the Sony MP-CL1, which has a zero offset and 42.1 degree horizontal FOV.

The Structure Sensor is mounted next to the projector with the center of its lens 20 mm below the projector lens such that its normal is equal to that of the projector.

4 Method

An overview of the correction method is presented here, and an elaboration the steps is provided below:

- Collect 320×240 depth image of scene without object surface
- Predict location of background and floor
- Loop:
  - Collect 320×240 depth image of scene with object surface
  - Perform necessary preprocessing on depth image
    - Remove back wall from image
    - Remove floor from image
    - Median filter image

- Calculate predicted object surface normal
- Calculate and apply homography on original image
- Render image

4.1 Back Wall and Floor Prediction and Thresholding

In order to isolate the depth image of the object surface, the location of the background and the floor must be predicted. This is a two step process in which the background is first predicted and removed, and then the floor is subsequently predicted and removed. Although the OpenNI interface used to collect data from the Structure Sensor already applies basic floor plane clipping, a substantial amount of additive noise remains in place of the floor which interferes with the surface normal calculation (see Figure 1). Thus, an approach [5] adapted from an article written by Petro Soininen and published on the Microsoft Developer Blog is used. This approach assumes that the background is close to normal to the image plane.

Figure 1: *Unprocessed depth image of scene without object screen. The image is normalized by max value. The black region at the bottom of the image is the floor, and the remaining grey region is the back wall. As can be seen, the noise on the floor plane takes the form of blotches with large values of uniform depth.*

In order to remove the background, a depth image of the scene without the object surface is obtained. Then, for this image, all of the non-zero pixels in the image are averaged. This average depth value is used as a prediction of the depth of the back wall. Subsequently, all pixels falling within a certain distance from this depth are thresholded out (set to zero). Note that this value is only computed once, and is used for processing on all subsequent frames.

Once the background is removed, a similar approach can be applied to remove the floor. This time, however, instead of averaging the depth coordinates, the vertical world coordinate of the pixels are averaged. This world coordinate (x_w, y_w, z_w) can be obtained from the image coordinates (x, y, z) as follows:

$$x_w = 2xz\tan(\frac{H}{2}) \tag{1}$$

$$y_w = 2yz\tan(\frac{V}{2}) \tag{2}$$

$$z_w = z \tag{3}$$

Figure 2: *Depth image with background filtered out. Floor noise in this image is even greater than that seen in Figure 1.*

where H is the horizontal FOV of the depth camera, V is the vertical FOV of the depth camera, and $x = 0, y = 0$ is the center of the image.

Once the world coordinates are found for each non-zero pixel, their y_w values are averaged, and the floor is predicted to be located at this predicted value. In subsequent frames, pixels that fall within a certain y distance from this value are thresholded out.

Figure 3: *Depth image with background and floor thresholded out. Some noise still remains in some frames due to the depth values being too high at those locations.*

4.2 Median Filtering

After thresholding to remove the back wall and the floor, although only pixels corresponding to the object surface remain, additional filtering must be applied. This is because there are still inaccuracies (salt and pepper noise, depth noise) in the depth measurements on the object surface. In order to address these inaccuracies, a median filter is applied to the thresholded image to filter out high frequency information in the image. Knowing that we only need the low frequency information on the location and orientation the flat object surface, we can use a fairly large median filter. For this project, a radius of 7 was used.

Figure 4: *Thresholded, median filtered depth image with poster board on which to be projected in scene.*

4.3 Predicting Surface Normal

In order to retrieve the orientation of the object surface, an approach similar to the one described in [6] is used. For a subsample of non-zero pixels in the image, a local normal is calculated. This is done by examining the world coordinates of the pixels above, below, to the left, and to the right of the pixel for which the normal is being calculated. Let u be the vector from the world coordinate of the left pixel to the world coordinate to of the right pixel. Let v be the vector from the world coordinate of the down pixel to the world coordinate of the up pixel. Then, the local normal can be computed as $u \times v$.

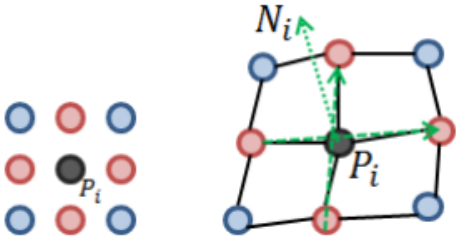


Figure 5: *Illustration of local normal*

Then, each calculated local normal is iteratively grouped into a plane. If no matching plane for a local normal already exists, then it begins its own plane. Two local normals N_i and N_j and their corresponding world coordinates P_i and P_j can be considered to belong to the same plane if the following two constraints are true:

$$|(N_i \cdot N_j) - 1| < \epsilon \tag{4}$$

$$|(P_i - P_j) \cdot N_i| < \epsilon \tag{5}$$

where ϵ is some small value (I use .01). These checks are extremely important for multiple different reasons. The first is that doing this check allows us to ignore occlusions of the object plane by arbitrary objects between the depth camera and the object plane. The second is that remaining pixels in the back wall or on the floor will be ignored by this check. Finally, noise in the depth image within the object surface that was not addressed by the median filtering will also be ignored.

After all local normals have been grouped into a plane, the group with the highest number of local normals is predicted to be the object surface. Then, all of the local normals for this plane are averaged to give the final value of the normal for the object surface.

The subsample for which local normals are calculated can vary, although for this application, the best results were shown for a subsample of pixels of width 3 lying on the two diagonals of the depth image. Other sampling patterns which were tested were a fully random subsampling, and a random subsampling of entire rows and columns of the image. A number of samples should be chosen which minimizes the amount of normals needed to be calculated while providing non-jittery performance. In testing, I used a subsample of size 1200 (about 1.6% of the pixels in the image).

4.4 Image Prewarping

After the object surface normal has been predicted, what remains is to prewarp the image such that it appears rectangular with the correct aspect ratio when viewed normal to the image on the surface. In order to do this, the object surface is simulated in an OpenFrameworks scene as an infinite plane. Then, the image is texture mapped onto the object surface with the correct aspect ratio. Finally, the virtual scene is rendered using a camera with the same intrinsics as the real life projector. Since the intrinsics for both the simulator camera and the real life projector are the same, the pixels for the desired image to be projected will land in real life where the texture-mapped image appears in the simulation (subject to scale).

In other words, let R be the projector matrix, which is equivalent to the inverse of the camera matrix $R^{-1} = C$. We simulate the object plane in the 3D scene, with the desired rectangular texture mapping. A point in the 3D scene $p = (x_s, y_s, z_s)$ is then rendered on the image to be projected at the homogenous coordinate Cp. Then, when the projector casts the image, since the normal of the simulated plane matches that of the real life plane, the distance of each casted pixel is proportional to that of the corresponding pixel in the simulated scene, thus the real world coordinates of the casted point p can be written as

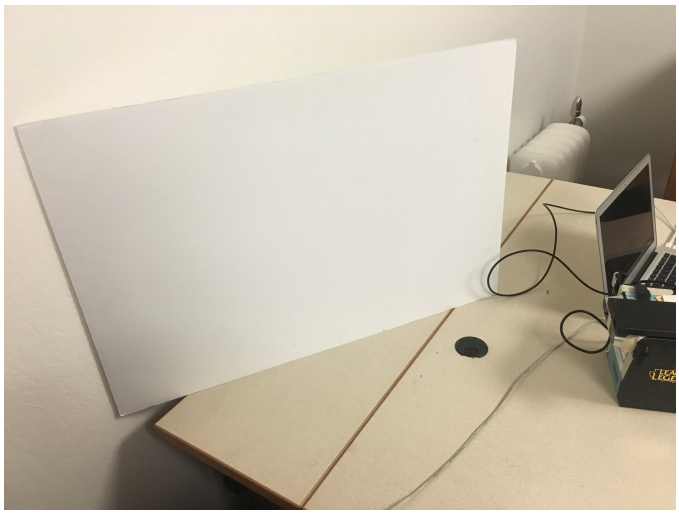
$$w = kRCp \tag{6}$$

$$= kp \tag{7}$$

where w is the real world coordinate of the casted point and k is a scaling coefficient.

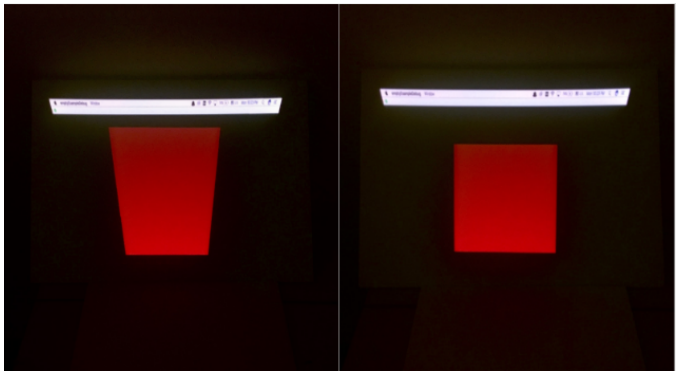
5 Experimental Results

Using this procedure with the Occipital Structure Sensor and the Sony MP-CL1, I was able to create a system which maintains a square image on a moving plane, as well as a system which casts a maintains image on a stationary wall while the projector and sensor move. Although the system performed well for small amounts of skew, it seemed to overestimate the incline magnitude for more extreme inclines. Furthermore, mixed incline (horizontal + vertical) was more likely to produce undesirable results at lower extremes since error in the horizontal and vertical direction are multiplied. Included along with the source code are videos of the tests which demonstrate the real time capabilities of this system. Below are images collected from the tests.

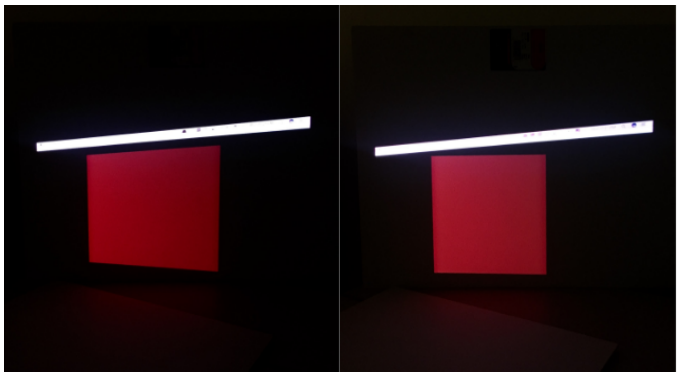


Picture of the set up. Posterboard used as object surface. Projector and depth camera at a fixed location with respect to each other with same rotation and small vertical offset.

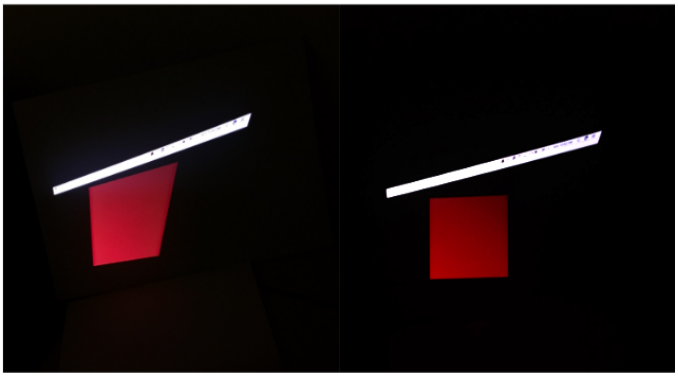
Original image to be projected.



Comparison of original vs. corrected casted projection onto a posterboard with vertical incline with respect to the projector. Both images taken by an RGB camera from a perspective normal to the poster board.



Comparison of original vs. corrected casted projection onto a posterboard with horizontal incline with respect to the projector. Both images taken by an RGB camera from a perspective normal to the poster board.



Comparison of original vs. corrected casted projection onto a posterboard with mixed incline with respect to the projector. Both images taken by an RGB camera from a perspective normal to the poster board.

Example of depth image in which the object surface is occluded by two bottles. Object surface is placed with a horizontal incline with respect to the projector.

Picture of projection produced from the previous depth image demonstrates how the occlusions do not affect the final predicted normal. Object surface is placed with a horizontal incline with respect to the projector.

6 References

[1] Agarwal, Anubhav, C. V. Jawahar, and P. J. Narayanan. "A survey of planar homography estimation techniques." Centre for Visual Information Technology, Tech. Rep. IIIT/TR/2005/12 (2005).

[2] Raskar, Ramesh, and Paul Beardsley. "A self-correcting projector." Computer Vision and Pattern Recognition, 2001. CVPR 2001. Proceedings of the 2001 IEEE Computer Society Conference on. Vol. 2. IEEE, 2001.

[3] Grossberg, Michael D., Harish Peri, Shree K. Nayar, and Peter N. Belhumeur. "Making one object look like another: Controlling appearance using a projector-camera system." In Computer Vision and Pattern Recognition, 2004. CVPR 2004. Proceedings of the 2004 IEEE Computer Society Conference on, vol. 1, pp. I-452. IEEE, 2004.

[4] Fujii, Kensaku, Michael D. Grossberg, and Shree K. Nayar. "A projector-camera system with real-time photometric adaptation for dynamic environments." In 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05), vol. 1, pp. 814-821. IEEE, 2005.

[5] Soininen, Pietro. "Background and Floor Removal from Depth Camera Data." Background and Floor Removal from Depth Camera Data. Microsoft, 21 July 2015. Web.

[6] Yoo, Hyun Woo, et al. "Real-time plane detection based on depth map from kinect." Robotics (ISR), 2013 44th International Symposium on. IEEE, 2013.