Francely Franco Bermudez

Christian Santana Diaz

Sheneeka Ward

Last edited: 7/31/2013

**Natural Feature Tracking for Rigid Objects on Augmented Reality Applications**

**Abstract**

One goal of AR research is to provide AR applications anywhere. Since AR relies on object tracking, this requires ubiquitous tracking of the user and physical objects. The problem that is being addressed in our research is Natural Feature Tracking (NFT) for rigid object tracking. Our question is: can we use NFT algorithms to track rigid objects? Recent implementations of NFT facilitate the robust tracking of 2D objects, such as photos, walls, tables, and newspapers. While this enables tracking of a wide range of objects, the physical environment is not entirely 2D. The novelty of our research is the systematic evaluation of all available feature tracking techniques.

-----ABSTRACT STILL INCOMPLETE------

# 1 INTRODUCTION

Augmented Reality (AR) technology is a Human-Computer Interaction technique that superimposes the physical world with virtual objects [1]. Key characteristics of AR are, 3D models that are aligned, spatial registered, with objects from the real world environment, and real-time interaction.

A goal of AR research is to provide AR applications anywhere. The idea behind this is that people will have another alternative like AR technology to solve their everyday problems. For example a teacher in a classroom can use AR to improve their student’s performance by guiding them through a series of steps that display virtual objects on top of real objects in order to complete an academic task.

Since AR relies on object tracking, this requires ubiquitous tracking of the user and physical objects. In general, three general tracking techniques exist: fiducial tracking, model-based tracking, and natural feature tracking. Fiducial markers are patterns that need to be attached on a surface in order to track physical objects. Using fiducial markers has its limitations because the environment must be prepared with markers. Model-based tracking utilizes 3D models of objects to track or the real word, which also limits the applicability of this technique. A prospective solution for this limitation is Natural Feature Tracking (NFT). NFT utilizes interest points in the environment (for instance corners, edges, and lines) to identify physical objects and to calculate camera position and orientation [2]. Common features that are tracked are corners, edges, and color gradients. . NFT needs photos from objects to track as reference. Since photo of the environment are easier to obtain than 3D models of the environment, this is a promising solution for holistic tracking.

The latest solutions of NFT are capable of tracking natural features on flat surfaces, like paper. AR Applications for mobile devices utilizes this functionality. The challenge is to promote this functionality to rigid objects. The tracking technique has been developed for 2D tracking, thus, the internal models as well as the descriptor matching algorithms are designed for 2D objects. These algorithms need to be more robustness. A robust algorithm in relation to feature tracking handles various inputs and will continue to function as expected.

In our research, we investigate the most robust feature descriptor for rigid object tracking. Recently, several novel feature descriptors have been introduced (i.e., BRISK [3], ORB [4], FREAK [5]). In comparison to mainstream feature descriptors like SIFT [6], SURF [7], Good Features to Track [8], and Optical Flow, they promising more robustness and performance. Our question is: can we use these descriptors to track rigid objects? Therefore, we have performed a systematic analysis of these feature descriptors and compared their tracking performance with the tracking performance of mainstream feature descriptors.

This paper is structured as followed. Next, the related work is introduced. In Section 3, we explain our method. This incorporates the hardware setup, the software, and the test method. Section 4 presents the results of our systematic analysis. We finish this paper with a summary and an outlook of future research.

# 2 RELATED WORK

Natural feature tracking (NFT) is a computer vision-based tracking techniques that utilizes distinguishable keypoints (e.g., color patterns) of an object to track in order to identify this object and to track it from video frame to video frame. The keypoints of a particular objects are extracted from that object in advanced and stored in a reference database. During application runtime, a video camera captures images of all objects, the subsequent algorithm extract the keypoints of all objects and their descriptors and matches them with all keypoint descriptors in the reference database. If the keypoints of the current frame can be matched with the keypoints in the database, the object is identified. Today, the feasibility of NFT techniques for 2D object tracking (i.e., photos, walls) has been proven. A huge amount of tracking techniques, programming frameworks, and applications are available. In 2005 Lepetit et al. [9] published a survey that provides an overview of the research in this area.

The research and the knowledge of 2D NFT tracking approaches for 3D rigid object tracking is limited. In general, the research in this field agrees that the robustness of 2D feature tracking techniques must be enhanced in order to facilitate 3D rigid object tracking. Two research works will be highlighted, which guides our research.  Han et al. [10] have introduced a key feature point tracking algorithm that utilizes prediction to enhance tracking robustness. They were able to predict and track key feature points by finding correspondences in two subsequent frames and by selecting good feature points. The algorithm results are that is inexpensive and robust to various type of motion but it is limited to non-real time.

Romero el al. [11] proposed a feature point tracking method taking into consideration the current color saturation when combining color invariance and luminance. The tracking is done by considering the color invariance for the colorspace and the corresponding luminance in a physical point located in an image. After, they make a list of key feature points and then sequences are played forward and reverse to verify if the points come back to their initial position. The results of their experiments demonstrate that more features are distinguished and tracked better regardless of orientation but the method is dependent of sigmoid parameters.

In Park et al. [12] the authors introduce a feature-based tracking system that incorporates model based object tracking and natural feature tracking. Their method combines object detection and frame-to-frame tracking. It distributes the time and complexity of detecting multiple objects over consecutive frames. From their experiments, it can be shown that multiple objects can be tracked with a reasonable frame rate. However, their approach still does not properly scale with the number of objects and it currently uses frame-to-frame tracking with no history and prediction techniques to store sufficient amount of data.

Most existing image-based methods for pose estimation either exploit textural information in form of local features or, if shape-based, rely on extraction of straight line segments or other primitives. The general problem is that a 3D object can potentially produce completely different 2D projections depending on its relative pose to the observing camera. Azad et al. [13], proposes a particle filter based tracking approach that can deal with arbitrary shapes and arbitrary or even no texture, i.e. it offers a general solution to the rigid object tracking problem. The core idea of the approach is to perform an appearance based matching based on online rendering of a finegrained local view space using a 3D object. For a global fine-grained view space, this would go far beyond reasonable memory consumption time. Therefore, they propose to render only those views that are in the vicinity of the current pose estimate, which has to be done online and this is all done within an annealed particle filter [21], which decides which object poses to render for each frame. In a different studies they mention that many real-time algorithms described in the literature still lack robustness, tend to drift, can lose a partially occluded target object, and are prone to jitter. To overcome these problems, Vacchetti et al. [14] developed an algorithm that merges the information from preceding frames in traditional recursive fashion with that provided by a very limited number of reference images, or keyframes. For a particular frame, the input data are correspondences computed using a fast technique that can handle both short and wide-baseline matching, this, can deal equally well with preceding frames, seen from relatively similar viewpoints, and keyframes whose viewpoints may be quite different. The approach demonstrated that it is faster and more accurate than the best current techniques. It is only slightly less accurate than one of the most recent iterative ones but much faster and more stable. Furthermore, the approach can be profitably be used to initialize the iterative algorithm.

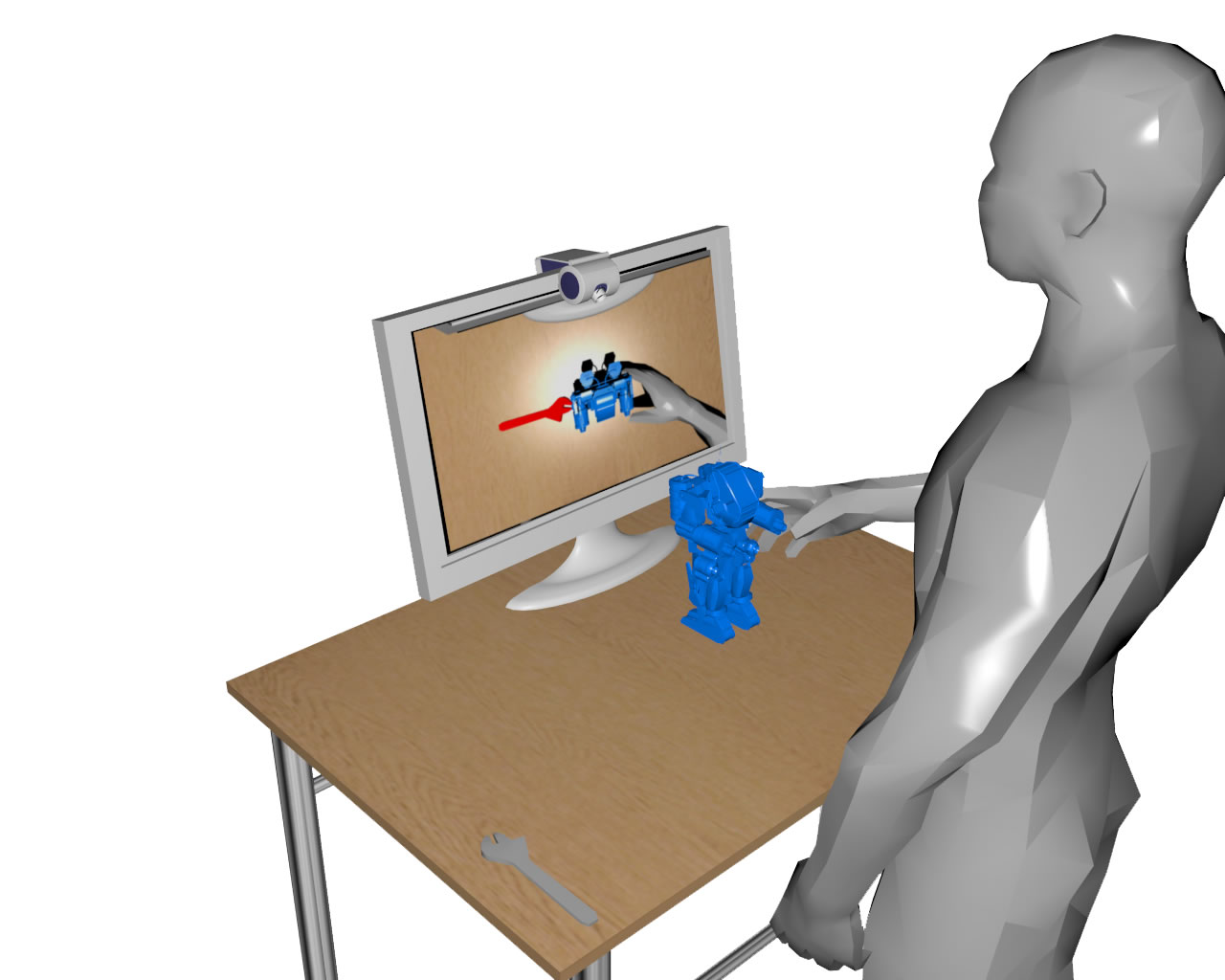
# 3 Systematic analysis of feature descriptors

The goal of this research is to investigate the feasibility of NFT for rigid object tracking. Since an accurate descriptor matching is mandatory for accurate pose estimation, we analyzed five different feature descriptors. The data show the accuracy and robustness when tracking rigid objects.

This section describes the systematic analysis of feature descriptors. First, we describe our hardware and software setup. Next, we explain the measurement of data

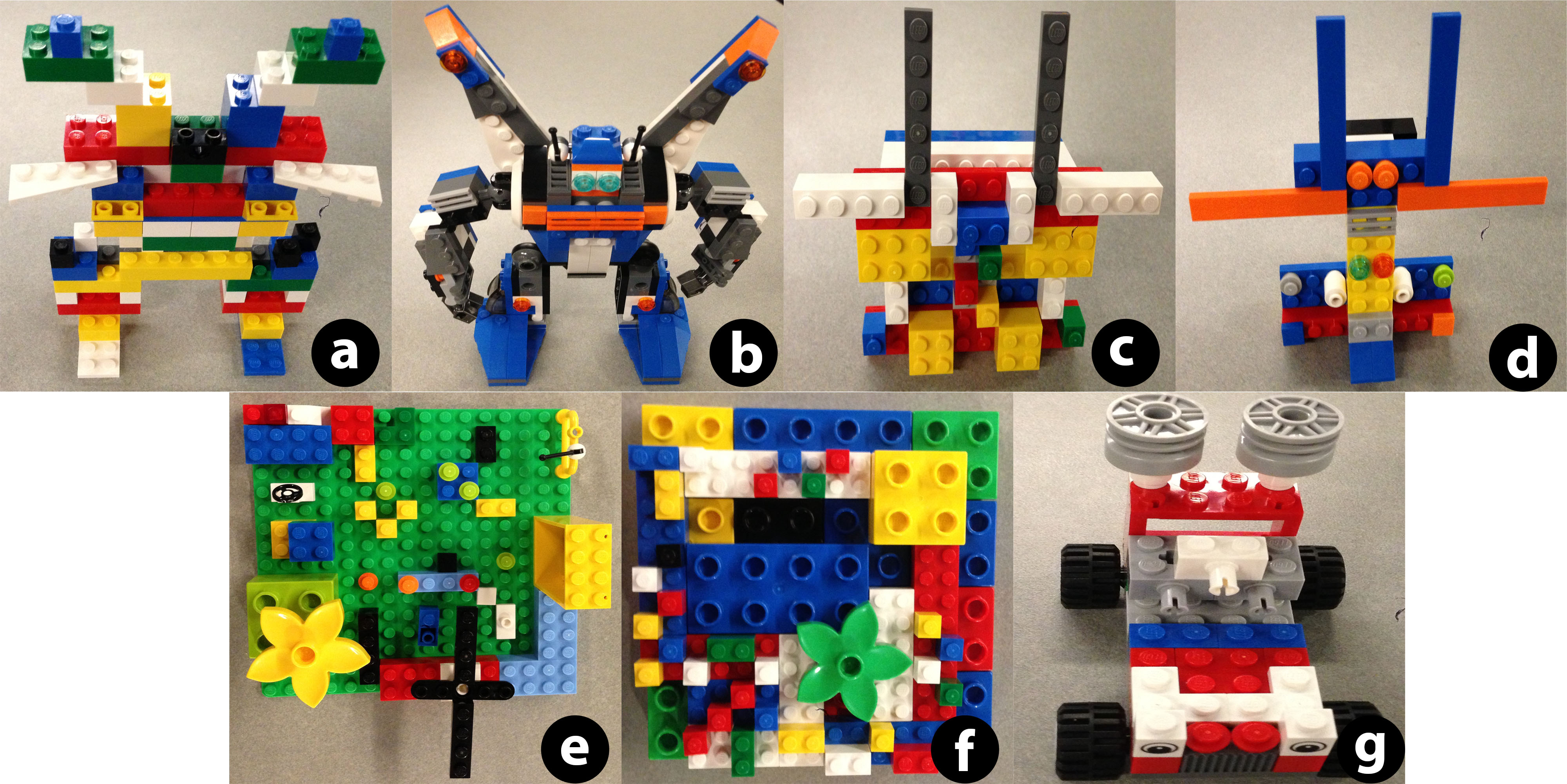
## 3.1 Hardware Setup

In Fig 1. we can see the hardware setup for our experiment. As hardware components we used a Dell Precision desktop computer with an Intel® Xeon® two processor at 3.60 GHz, 12 GB of RAM and an NVIDIA Quadro 6000 graphics card. Also our setup incorporates a table top application with a Creative 1.3 MP Webcam and a monitor as output display.



**Figure 1. The hardware setup for our experiment.**

What is the difference between them?



**Figure 2. Seven rigid objects to track.**

Fig 2. shows seven physical objects to track, designed using Lego bricks. All objects alter in size, color and spatial structure. We designed our own rigid objects using Lego bricks because it will let us to hold control over the size of the objects to test if there is a significant difference when matching features. In addition, the variation of colors in the Lego bricks allows an easier way for the feature detector to do its work by finding color contrast in the object. Finally, the spatial structure of the objects were arranged to create a vast amount of good features to track.

DEFINE SIZE OF THE OBJECT STILL INCOMPLETE

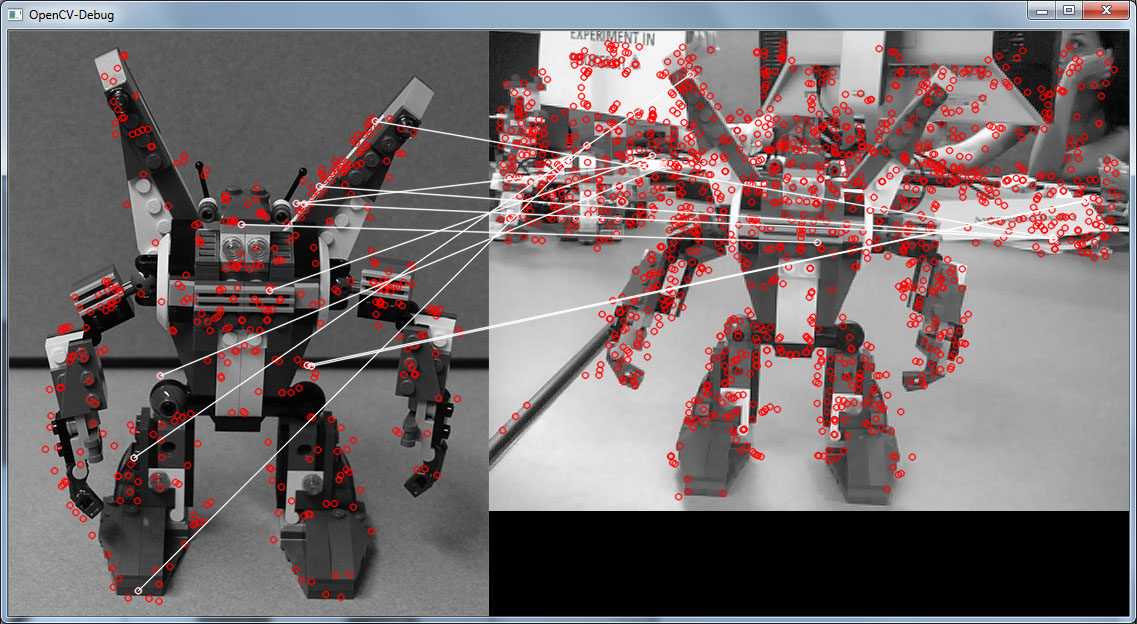
Objects a and b are very similar in size but different in spatial structure. In comparison with the rest of the objects they are consider the big objects for our experiment. Objects c and d are the medium size objects because they are similar in size but different in color and spatial structure. Objects e and f are both wide in space and small in height but are different slightly different in spatial structure.

---------------------------------------------------------------------------------------------------------------------

## 3.2 Software

First, draw a diagram that shows the processing of the camera images. How do you get your data?

The software that we use to overlay virtual images over the physical world is ARToolkit. ARToolkit is a library that is used to build Augmented Reality applications. For our tracking and matching we have been using descriptors and detectors from the library found in OpenCV. OpenCv is highly efficient for computational processes in real time applications.



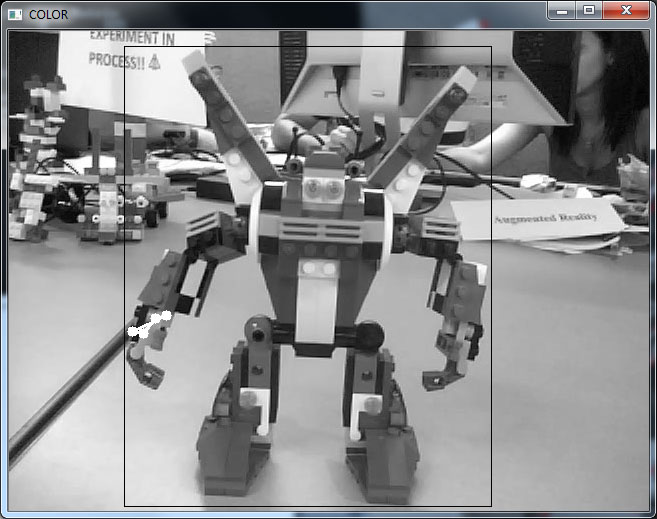
**Figure 3. This are the matches from the picture database to a rigid object using OpenCV.**

## 3.3 Procedure

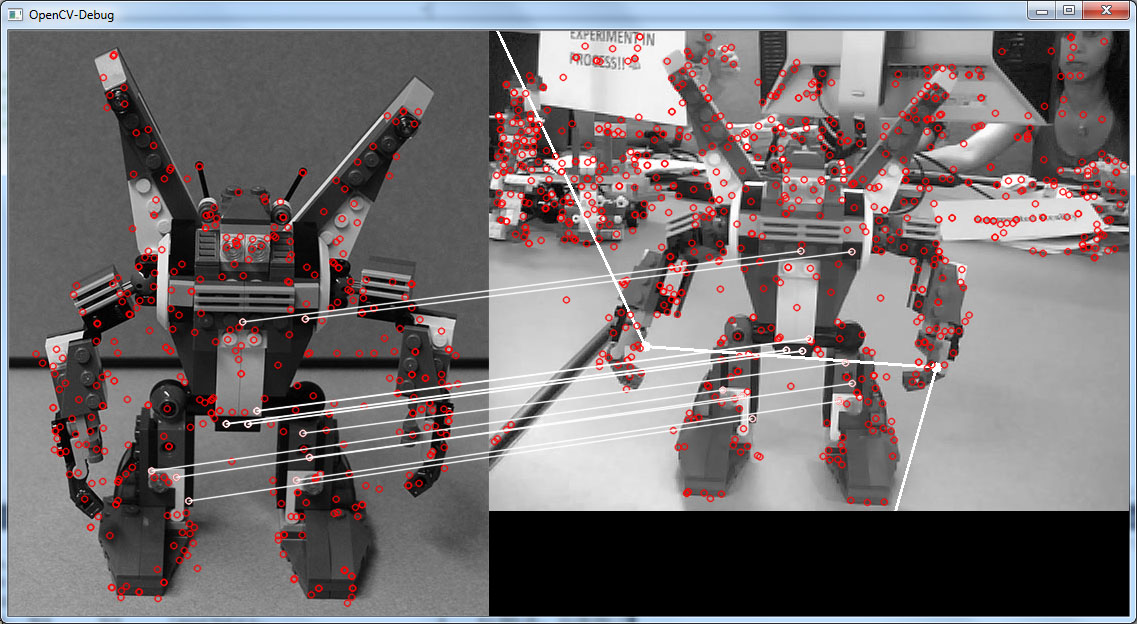
First, describe the entire procedure.

Second, tell your reader which data you measure. Use English sentence for that and not only English words in an awkward order.

Third, add some figures that exactly explain how you get your data. For instance, explain the region of interest. Show in this image what you consider as a true positive match and what you consider as a false positive match. And explain this.



**Figure 4. Shows the region of interest on a video of a rigid object.**



**Figure 5. Shows true positive matching.**

# 4 EXPERIMENTAL RESULTS

This section presents the results of the experiments. The first subsection explains the analysis. We describe the math to obtain results out of our raw data. The next subsection shows the results. Next, we discuss them and finish this section with a conclusion.

## 4.1 Analysis

You are able to add two things to this section.

First, add the math that explain how you get from your raw data to your results!!

## 4.2 Results

Second, add the table / diagrams that show the results. Even if you have no data to describe, you are still able to add the diagrams / tables, add some dummy data and describe the columns and rows of the table and the axis of the diagrams.

## 4.3 Discussion

Discussion and conclusion need to be empty

## 4.4 Conclusion

# 5 FUTURE WORK

# 6 ACKNOWLEDGEMENTS

This research was performed at Iowa State University as part of a research internship program for Interdisciplinary Research and Education and Emerging Interface Technologies [SPIRE-EIT] during summer 2013.We also acknowledge the National Science Foundation for financially supporting this project through NSF Grant CNS-1156841. (AKNOWLEDGE ATUTHORS)