Aim Point Visualization Using ArUco Markers

Shrek Strickland
CPSC 4820 - Computer Vision
Clemson University
Clemson, SC

April 18th, 2021
Email: jstric7@g.clemson.edu
ID: C20193462

Abstract

One of the last projects we completed in the semester was the detection and rudimentary implementation of ArUco markers to achieve an augmented reality style of effect. Since augmented reality is currently a very big topic in the field of computer vision, I decided to delve further into this concept with a practical use. This paper serves to document my results in creating an implementation of OpenCV's ArUco detection and image transformation in order to create aim lines for firearms, in order to create a viable alternative to laser sights.

Keywords: OpenCV, aim point, visualization, augmented reality, ArUco markers

---

1. Background

When first starting this class, there were several topics we covered, such as Haar filters, calibration matrices, and even face detection. However, no topic resonated as quite as much as the ArUco marker lessons did. The idea that any object can have its pose and location tracked with a simple sticker was a very versatile idea to me, and I was immediately thinking of possible project ideas. Robots could use it for object recognition and handling, VR headsets could use them to automatically determine boundaries of an area, but I think the most significant field where they shine is augmented reality.

As someone who likes to keep my projects practical instead of purely theoretical, I began thinking about fields that could benefit from this technology. Different ideas for things like wrist HUDs were interesting, for sure, but they weren't practical. Then the idea to use them for military purposes hit me.

Laser sights, though they may make it way easier to aim, have the major flaw of making the user very visible from quite a distance away. Special ops units have a workaround in the form of infrared sights, but these can only really be used in certain circumstances, such as night time. My idea, therefore, was to use markers on weapons in order to shoot out a laser virtually, passing it into something like a Google Glass or equivalent headset.

While I do not actually own a headset of that variety, and therefore cannot program something to work with it, I created a version that simply edits a video as a sort of proof of concept.

2. Methodology

2.1. The Weapons of Tomorrow

The first hurdle I had to overcome in the creation of this program was that of coming up with a design that could satisfactorily represent the firearm with its respective marker. I had several ideas for this, such as affixing two markers to it in order to find a vector between the two, creating a forward vector. However, I soon realized that only one marker would be needed, as each marker carries with it orientation information. Therefore, as long as the marker's orientation lined up with that of the barrel, only one marker would be needed. So, I created a cardboard replica of a gun and affixed an ArUco marker to it.

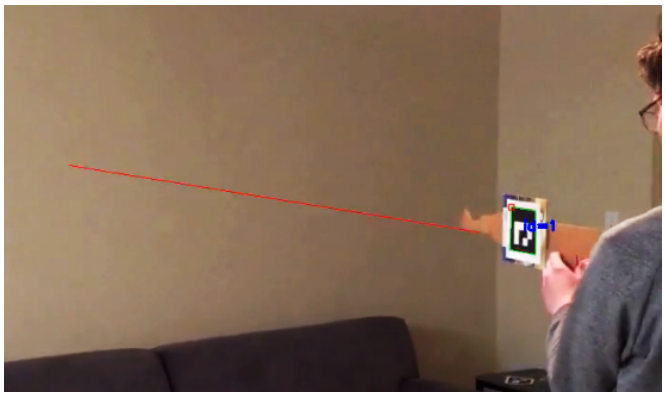

Figure 1 - A depiction of the resulting aim line

The method by which I went about detecting the gun's marker and grabbing the orientation information was a rather simple one, detailed in the OpenCV documentation [1]. The basis of the idea is that the algorithm is given a predefined set of markers to look for, generates a very high contrast version of the image in order to help isolate the markers, and then searches the resulting image for matching patterns. Additionally, a real-world size (in meters) is provided for the markers, allowing the algorithm to determine how far away it is from the camera

along with calculating the orientation. It is from this data that I was able to get the forward vector I would need to use to orient the aim line.

2.2. Marker, Marker, on the Wall

The next improvement I made to the implementation was the placement of a secondary ArUco marker on the wall that the gun was aimed at. This was necessary in order to obtain accurate depth information for the object being aimed at, and provides much needed information for future developments, such as ricochet angle. The most practical use, however, is to use such information with 3D intersection algorithms [2] that enable a more accurate aim line.

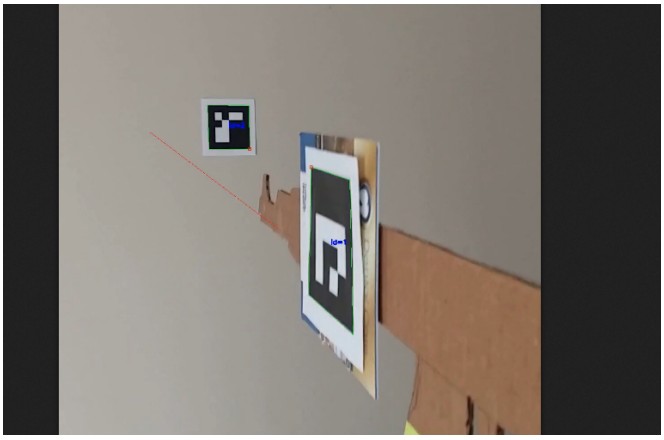

Figure 2 - The wall marker provides a reference point

3. Future Improvements

3.1. Reflection Vector Ricochet

One interesting application of a computer vision-based aim line instead of a physical one based on lasers or infrared data is that many more things can be done with the dataset obtained in the process of line drawing. For example, ricochet analysis. Either for safety or combat purposes, ricochet analysis has several advantages, but would be quite complex to implement. Similar to reflection vectors used in graphics programming, ricochet vectors would heavily rely on the materials of the object struck, in addition to the specifications of the gun. This would require a large dataset and complicated algorithm that I don't have access to and don't have the time to create.

3.2. Aim Training

Another interesting field to delve into would be aim training. By using the positional info from the marker, data could be collected on the way a soldier controls the gun while firing, or even the way the gun moves while firing, helping a soldier improve his aim, or providing data to a manufacturer to help improve a firearm.

4. Lessons Learned

At the beginning of the semester, I had plenty of experience in computer graphics. Ever since my introductory C course, I have loved image manipulation through code. I took both the 3D and 2D graphics courses here in the same semester, and my final project for my GPGPU class was data visualization. However, even with all of this graphics experience, I was still almost clueless to most topics in computer vision. Thinking it was too difficult to handle for someone like me, I just ventured away from the field. However, through this class, I realized just how accessible it has become thanks to these advanced libraries. And even though the library makes it simple, I still can realize just how much complex math goes into each function.

References

[1] “Detection of ArUco Markers”. OpenCV-Tutorials-for-contrib-modules, https://docs.opencv.org/master/d5/dae/tutorial_aruco_detection.html

[2] “Line and Plane Intersection (2D & 3D)”. Geomalgorithms, http://geomalgorithms.com/a05-_intersect-1.html

[3] “Ricochet Analysis Introduction”. Bev Fitchett’s Guns, March 27, 2021, https://www.bevfitchett.us/ballistics/ricochet-analysis-introduction.html#:~:text=Ricochet%20Analysis%20Introduction%20%20%20Calibre%20%20,%20%201.7%20%2013%20more%20rows%20