ChromaTag - A Colored Fiducial Marker

Austin Walters
Department of Computer Science
University of Illinois Urbana-Champaign
awalte9@illinois.edu

Bhargava Manja
Department of Computer Science
University of Illinois Urbana-Champaign
manja2@illinois.edu

Abstract

Though one of the major goals of computer vision is automatic recognition and understanding of natural features, for many applications this level of sophistication is unnecessary. In applications where perception is not the major focus, such as virtual and augmented reality systems and swarm robotics, localization and ground truthing is achieved with the use of fiducial markers, which serve as easily identifiable artificial features. In this paper we describe ChromaTag, a drastic improvement on a widely used and researched fiducial system called AprilTag. Our system improves upon AprilTag by decreasing the computational cost of tag detection and increasing robustness to lighting, occlusion, and warping. We achieved this, in part, by eschewing the standard RGB colorspace in favor of CIELab or YUV and by incorporating color into AprilTag. As a result, the detection rate of ChromaTag is nearly twice that of AprilTag, allowing use in real time applications. We demonstrate these results and highlight directions for future experimental work.

1 Introduction

Fiducial markers are artificial features designed to work as unique identifiers. That is, they are designed to be easy to localize and understand even in suboptimal conditions (occlusion, poor lighting, etc.) and to hold small amounts of identifying information. Thus, though they often look like QR codes, they are fundamentally different in design and application. QR codes require human alignment of the image and a high resolution image to identify large information payloads (enough to contain URLs). Fiducial markers contain far less information, but must be identifiable even when arbitrarily located or rotated within an image, and under arbitrary lighting conditions and image resolutions.

Fiducial markers have applications in a wide variety of fields. They are used extensively in augmented reality applications, where they are put on real world objects so that augmented reality systems can superimpose graphics onto them. These applications spawned the first few widely used fiducial marker systems, such as ARTag and ARToolkit. Fiducial systems are also used in many disparate ways in the field of robotics. Fiducial systems have been used to design human/robot interaction systems that are as simple as flashing tags to communicate with machines. One particularly important application is in the performance evaluation of robotic systems. Fiducials can be used to generate ground truth trajectories and evaluate SLAM algorithms.

In this paper, we describe a drastic improvement on a particularly widely used fiducial system called AprilTag. This system is a significant improvement on previous systems, both in terms of performance and for the fact that it was released under a free and open license. AprilTag provided significantly better localization accuracy and robustness to rotation, occlusion, and false positives. However, by extending AprilTag's coding system to include color, we can improve in all of these areas, as we will discuss.

2 Previous Work

AprilTag was developed by Edwin Olson from the University of Michigan[1]. AprilTags consist of a black and white square with a binary encoding inside. AprilTags work by reviewing gray scale images to attempt to find the tag.

Figure 1: AprilTag 36h11 ID = 12

These tags are tracked and identified using a relatively simple process. We decided to use the same exact method for our intial testing of ChromaTags, enabling a more direct and accurate comparison.

AprilTag Method

Line Detection
The first step in the AprilTags identification method is to compute the gradient and magnitude of every pixel, enabling the detection of line segments using efficient methods such as those described by Felzenszwalb[2].

Quad Detection
After all of the line segments have been identified, the AprilTags algorithm completes a depth-first search on the line segments. If four lines intersect, then they are a potential quad and need to be checked for detection.

Homography and Extrinsics Estimation
After a quad is detected the homography is determined using the Direct Linear Transform (DLT) algorithm[3]. Then, with the cameras focal length, it is possible to solve the PnP problem using RANSAC, providing us with a rotation and translation vector.

Payload Detection
Once the rotation and translation is known it is possible to check the payload of the quad. Using the homography the inside of all of the potential quads are sampled and determined to either be a known tag or not a tag.

One of the more important insights by Olson was the development of the AprilTag coding system. It was designed to[1]:

1. Maximize the number of distinguishable codes
2. Maximize correctable bit errors
3. Minimize false positive confusion rate
4. Minimize the number of bits per tag (minimizing tag size)

This was accomplished by maximizing the Hamming distance between the tags, thus making them more distinguishable. Critically, the Hamming distance from all other tags are bounded from below, no matter how the tags are rotated (i.e. 90, 180, 270 degrees).

Based off experimental data, AprilTags was a significant improvement over prior methods such as QR codes or the ARToolKit[4]. Notably, AprilTags are significantly better at reducing false positives, specifically those which occur due to either part of the tag being covered, lighting issues, or errors due to color. Essentially, the algorithm will identify a tag incorrectly or may not identify it at all, which can be catastrophic in instances using robots, or break an augmented reality experience.

For reference, a 6x6 AprilTag can correct 3 bits and still have a false positive rate less than one percent of the time.

Bits corrected	36h10 FP (%)	36h15 FP (%)
0	0.000001	0.000000
1	0.000041	0.000002
2	0.000744	0.000029
3	0.008714	0.000341
4	0.074459	0.002912
5	0.495232	0.019370
6	N/A	0.104403
7	N/A	0.468827

Figure 2: Bit correction error rate for different tag families

3 Why ChromaTag

ChromaTags currently are still in development. However, we have finished a large portion of the theoretical work and have recently developed an initial tag for comparison purposes with AprilTags.

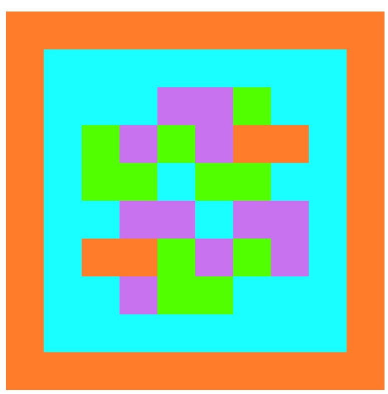


Figure 3: ChromaTag 36h11 ID = 12

The goals behind developing chromaTags are to:

A. Decrease tag idenfication speed
By decreasing tag identification time it will improve the trackability of the tag in realtime. For example, improving tag tracking from a few frames per second to fourty or fifty will make a drastic difference in an augmented reality experience.

B. Reduce false positives
In Olson's AprilTags paper, it was noted how important reducing false positives can be. Any improvements with this in mind would improve the robustness of a tag.

C. Minimize the size of a tag
AprilTags are often the size of a sheet of paper, roughly 15cm across. Developing

a tag that is 3cm across would be a dramatic reduction in size, and would enable a wider range of applications.

D. Maximize the number of codes

Given the requirement for every AprilTag to have a hamming distance greater than a certain number (ideally 10 or 11), it severally limited the number of tags. Rather than having only a few thousand tags, we desired to increase that number to millions.

With those goals in mind, we decided to take a somewhat different approach than previous tag methods. In computer vision it is common to use either edges or blobs for identification purposes, usually using a gray image, since it has three times less data. Unfortunately, there are often many edges in a scene, perhaps thousands, and every edge does not always provide important data.

For example, the AprilTags algorithm searches the scene for every gradient in a gray scaled image, then generates line segments. This method produces an incredible amount of edges, the overwhelming majority of which are meaningless. The downside is that it is not possible to effectively identify AprilTags without searching the entire scene and taking the gradient of every pixel.

An alternative to gray scale, is of course RGB. Using RGB would allow many of the edges in a scene to be ignored. For example, if we only found the gradient in the red (R) channel, edges in the blue and green channels could be completely ignored, reducing some of the edges we needed to check.

However, RGB contains light information. For instance, red in a light room will look very different from the same exact red in a darker room. Thus, using RGB would not be advised as it would likely be less robust to lighting conditions than gray scale.

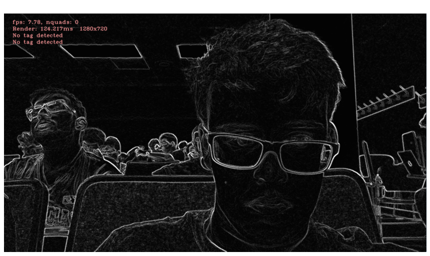
Upon further inspection, other color spaces such as YUV or CIELab (henceforth refered to as Lab), which separate out the light channel, could be used to great effect, removing light all together. Further, other color spaces enable a gradient to be more easily found, especially in Lab.

In the Lab color space the

L channel represents the light channel
a channel represents colors green to red
a channel represents colors blue to yellow

When finding the gradient of every pixel in the a channel, green and red rarely appear next to each other in a scene and thus very few edges will appear.

For a visual comparison:


(a) Edges of a gray image


(b) Edges of the a channel Lab image

The number of edges in an image are greatly reduced, which provides a significant speedup, and provides a more robust tag which is less effected by light (since Lab separates out light as a separate channel).

Finally, by using two color channels (a and b) as opposed to one (gray), the amount of information that can be stored per tag has grown by a power of two. This implies that a ChromaTag can be significantly reduced in size, but will also be more robust to false positives, as we will show in our method.

4 ChromaTag Method

Currently, chromaTags use the same exact algorithm as AprilTags to identify and track the tag. Although this is not optimal, it is a direct way to compare the tags, to highlight advantages of chromaTags. With this in mind, the only step(s) that are required to identify and track chromaTags are preprocessing step(s).

Further, by constraining ourselves to using the AprilTags algorithm for testing, we have reduced the need to create a whole new set of tags. At this stage, chromaTags are created by taking a standard 36h11 AprilTag and placing them on the same chromaTag.

ChromaTag Creation Method

1. Convert an AprilTag to the b color channel of the Lab color space
2. Convert an AprilTag to the a color channel of the Lab color space
3. Rotate one tag 180 degrees
4. Place both new colored AprilTags ontop of one another and blend colors

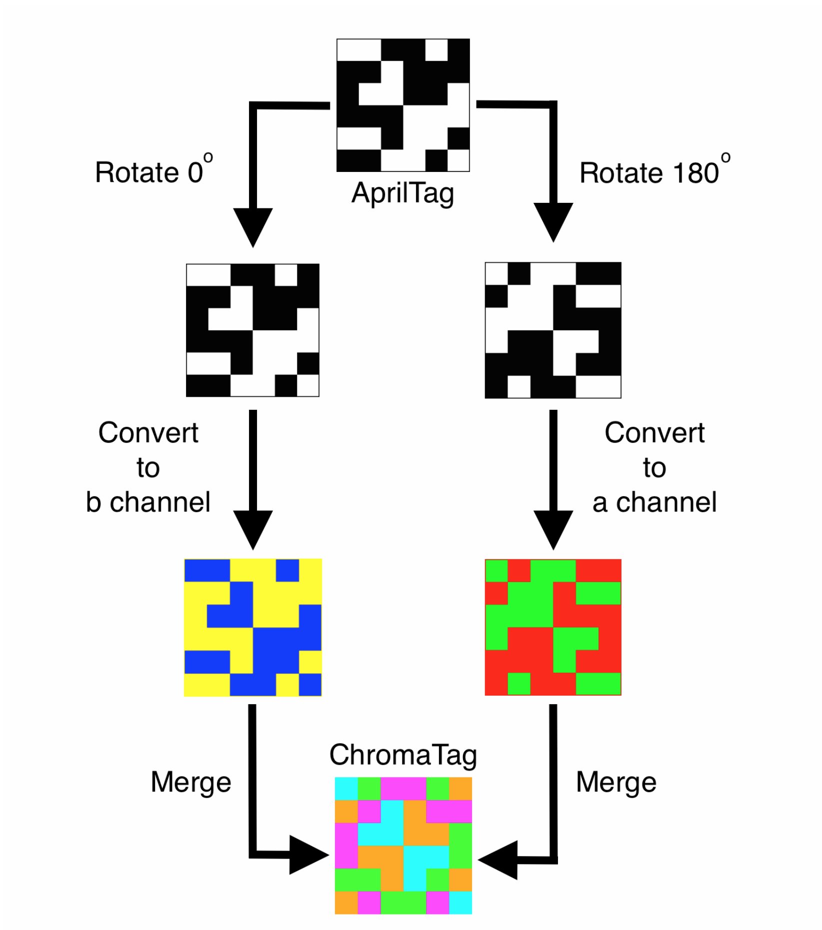


Figure 5: A graphical depiction of the ChromaTag method

Thus, the same AprilTag or two different AprilTags can be encoded twice on the same chromaTag (generation code is available on our github [5])!

It is important to note, that this tag generation is not achieved by simply mixing colors in the standard RGB color space, that would not achieve the appropriate colors. Rather, the colors we select for the tag are at the corners of the Lab color space, such that the a channel and b channel would both be at the extreme when selecting a color, enabling a high gradient to be achieved.

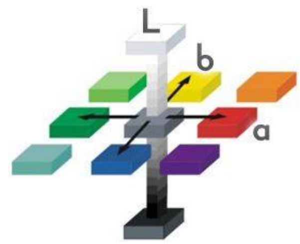
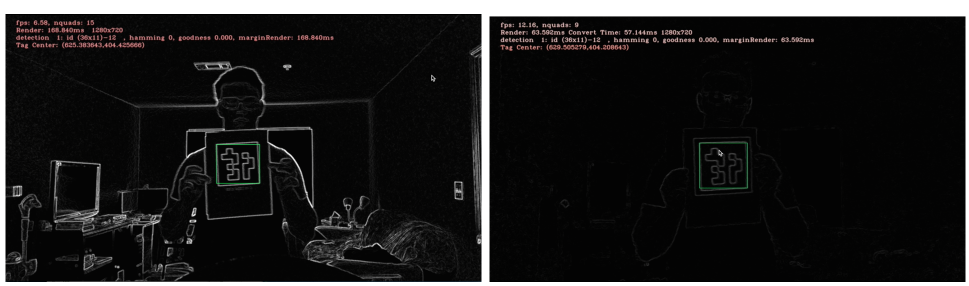


Figure 6: The Lab color space

After creating a chromaTag, idenfication only requires three additional steps.

ChromaTag Identification Method

1. **Conversion from RGB to Lab**
   This is a relatively simple step, and we achieved it using OpenCVs converstion BRG2LAB function.
2. **Seperating of a and b channel**
   Once we have an Lab image, we can seperate out the a channel and then hand the new frame to the AprilTags algorithm. The a channel tags are more prominent, and fewer superfluous line segments are discovered, so that is the first to be sent to the AprilTags algorithm.
3. **Run AprilTags algorithm again**
   Initially the AprilTags algorithm is ran for one color channel (the a channel). After a tags location was identified, then the algorithm is ran again over a very small localized region for the b channel, and that tag ID is also identified.



(a) AprilTag detection

(b) ChromaTag detection

Even though the AprilTags algorithm runs twice, and every frame needs to be converted from RGB to Lab, there is a significant reduction in processing time for each frame. This speed up is due to the reduction in number of superfluous line segments that are checked by the AprilTags algorithm (as shown in the figure above).

5 Initial Results

There are a significant number of improvements that can be made to our method, however initial results are very promising. Though, due to time constraints, we have not been able to more completely experiment and compare the performance differences between AprilTag and ChromaTag, or to determine the performance characteristics of ChromaTag, initial testing has shown significant speedup in determining tag location and orientation. Our experimental setup involved measuring the detection rate and measuring the maximum detection distance of an AprilTag and the equivalent ChromaTag. We used the AprilTag C codebase as a reference implementation and based our ChromaTag implementation off of it, using the same algorithms and data structures, which provides for a fairer comparison. We used OpenCV to do image capture and RGB to Lab conversion. Our testing benchmarked AprilTag as having a framerate of roughly 7 frames per second on a 2013 15 inch MacBook Pro, slightly below the threshold required for real time usage, and even close to the promised real time performance (which, according to the AprilTag website, is achievable on a cellphone processor). ChromaTag, with and identical setup, clocked in at 14 frames per second, double that of AprilTag. This too with an unoptimized implementation that does expensive RGB to Lab conversion per frame. In addition, we found that ChromaTags were detectable at twice the distance, even with suboptimal lighting conditions (bright lights that caused the capture camera to recalibrate).

6 Future Work and Conclusions

As previously mentioned, there is quite a bit left to be done in exploring the usefulness of ChromaTags, specifically in comparing it to AprilTags. We plan on continuing this exploration experimentally. One important measure is robustness. Though we know ChromaTag is more robust to occlusion and lighting conditions than AprilTags, we need to quantify this improvement to determine where ChromaTag can not be used.

Another important metric is robustness to bit detection errors. The power of the AprilTag scheme is in its error correction properties, and the addition of color may make error detection and correction even more robust. Positive results in this direction would drastically increase the useability of fiducial systems in wide varieties of suboptimal conditions, such as, for example, rescue robotics or mission critical vision systems.

Though we have demonstrated marked improvements by using the Lab colorspace, other colorspaces (or even a custom colorspace) may make quad detection rates faster or increase robustness to certain types of conditions. Most of our work will involve exploring different colorspaces and finding out how they impact performance.

Another goal is to move away from using AprilTags' basic algorithm, as changing colorspaces presents many opportunities for optimization. For example, the fact that certain types of edges can be thresholded away in Lab space makes possible the use of better algorithms for quad detection and line fitting. This thread of inquiry presents might present major advancements in the state of the art of fiducial systems.

Though there is still work to be done, we have taken the first steps on the path to dramatically improving fiducial systems. We have shown that a simple series of pre and post processing steps can dramatically improve upon AprilTag, and have highlighted areas for further improvements.

7 References

[1] Olson, Edwin. "AprilTag: A robust and flexible visual fiducial system." Robotics and Automation (ICRA), 2011 IEEE International Conference on. IEEE, 2011.

[2] P. F. Felzenszwalb and D. P. Huttenlocher, "Efficient graph-based im- age segmentation," International Journal of Computer Vision, vol. 59, no. 2, pp. 167–181, 2004.

[3] R. Hartley and A. Zisserman, Multiple View Geometry in Computer Vision, 2nd ed. Cambridge University Press, 2004.

[4] D. Wagner, G. Reitmayr, A. Mulloni, T. Drummond, and D. Schmal- stieg, "Pose tracking from natural features on mobile phones," in IS- MAR '08: Proceedings of the 7th IEEE/ACM International Symposium on Mixed and Augmented Reality. Washington, DC, USA: IEEE Computer Society, 2008, pp. 125–134.

[5] github: https://github.com/lettergram/chromatag