A Motion Blur Resilient Fiducial For Quadcopter Imaging

Meghshyam G. Prasad & Sharat Chandran
Dept. of Computer Science and Engineering, IIT Bombay
{meghshyam,sharat}@cse.iitb.ac.in

Michael S. Brown
School of Computing, NUS
brown@comp.nus.edu.sg

Abstract

Fiducials are commonly placed in environments to provide a uniquely identifiable object in the scene. In quadcopter applications, these fiducials are often used to evaluate planning algorithms given that ground truth positions can be detected from the quadcopter's camera.

Low cost quadcopters, however, are subject to quick and unstable motions that can cause significant motion blur that severely affects the detection rate of existing fiducials. This problem motivated us to design a fiducial that is robust to motion blur. Our proposed design uses concentric circles with the observation that the direction perpendicular to the motion blur direction will be relatively unaffected by the blur. As a result, an appropriate fiducial code orthogonal to the blur direction can be recognized. Since the direction of motion blur is unknown, the circular design is good for all motion blur directions. We describe the design of binary fiducials, and also a detection algorithm. We show that our marker can significantly outperform existing fiducials in scenes captured with a quadcopter.

1. Introduction

The recent availability of low-cost quadcopters has helped fuel significant efforts in research focused on unmanned aerial vehicles. Navigation and planning of these vehicles is typically [10, 11, 12] performed using onboard inertial sensors and/or vision based modules that uses visual cues in the real world. However, to evaluate the effectiveness of navigation methods, one or more fiducials are commonly placed [7, 18, 16] in the environment to provide additional information that serves for ground-truth positional measurements.

In order to be effective, fiducial markers (or simply, fiducials) need to be easily detected in the scene. A variety of fiducials have been proposed [20, 1, 13, 5, 4] in the literature. These take the form of binary codes arranged into rectangular grids [1, 13] or other geometric primitives arranged in predefined spatial patterns [20, 5, 4]. Figure 1 shows the popular ARTag [13] fiducial as possibly seen from a quadcopter. A problem for existing fiducials is that low-cost quadcopters often exhibit very quick and erratic physical movements that result in motion blur evidenced in the images captured from the quadcopter's onboard camera. This motion blur has an adverse effect on the recognition of fiducial markers. This can be seen in Figure 1–(b) where the ARTag fiducial cannot be recognized due to motion blur. This is not too surprising as most existing fiducials are not designed to handle motion blur.

Compounding this problem is the additional issue of dropped video frames from the quadcopter's wireless communication module. This means that not only is blur a problem, but there may be large discontinuities in the pattern's position due to missing video frames. As we will show in Section 4, this later problem makes it challenging to apply tracking algorithms that can exploit temporal coherence for determining the fiducial's position.

Contributions: To address these problems, we propose a fiducial that is designed to be resistant to motion blur. Our design is based on circles as shown in Figure 1–(c,d). The design is based on the observation that motion blur from a quadcopter tends to be linear in nature. As such, when our fiducial is blurred, there is no blur in the direction perpendicular to the direction of motion. This allows the signature of the fiducial to remain intact in any direction.

When multiple fiducials are present, using concentric rings, we can treat the presence or absence of a ring as a bit, allowing us to assign a code to the marker. Our experiments show that these designs can significantly outperform existing codes in the presence of motion blur. As far as we are aware, this is the first work to propose a blur resistant marker especially in quadcopter settings.

The remainder of this paper is organized as follows. Section 2 gives an overview of related work in fiducials, as well as the related problem of tracking. Section 3 motivates our fiducial design by analyzing the performance of existing codes under motion blur, and describes our detection algorithm. Section 4 shows several experiments using quadcopter imagery. This is followed by a discussion and a summary in Section 5 and Section 6 respectively.

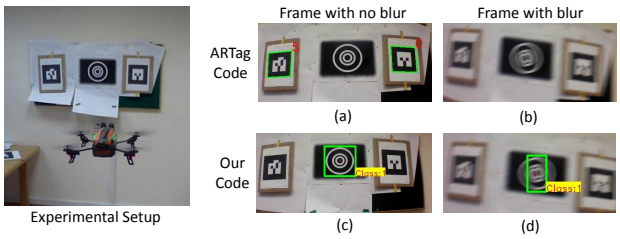


Figure 1: Experimental setup and comparison of fiducial-based algorithms. A green border signifies success. **(Left)** A drone encounters three different fiducials. **(Right, Top)** Output of ALVAR [26] under favorable and blurred circumstances. No green border is seen in (b) signifying failure. **(Right, Bottom)** Output of the fiducial proposed in this paper. Green border is seen in both (c) and (d).

2. Related Work

Our work is related to two areas: fiducials and object tracking. We briefly discuss work done in both areas.

Fiducials: Figure 2 shows a few examples of existing fiducials. Many designs use a two dimensional barcode inside a rectangular grid. One example of such a fiducial is from the ARToolkit [1], a well known toolkit used in many augmented reality (AR) applications. Kato and Billinghurst [15] first demonstrated the use of ARToolkit in various augmented-reality-based applications.

Fiala [13] proposed a fiducial termed, ARTag, which is a bi-tonal system consisting of a square border and an interior 6×6 grid of black or white cells. The improvement of the ARTag compared to the ARToolkit lies in the detection of corners instead of detection of lines to find possible patterns. This proved to be more efficient than [1] in terms of recognition rate as well as the number of different patterns which can be created. *The reliance on both line and corner detection, however, hampers recognition under motion blur.*

There were also attempts to use circular patterns instead of rectangular. Gatrell et al. [14] used concentric circles for monocular pose estimation as well as object identification. Cho et al. [8, 9] have used multicolor rings instead of black and white rings [14] to increase possible number of fiducials. These multicolor rings are used in wide area tracking in large scale applications. *Although based on concentric rings, these approaches require the complete ring to be recognized; this is impractical when the pattern undergoes directional motion blur.*

Figure 2: Prior fiducials and our proposal. While each code has its pros and cons depending on the environment, no prior code is expressly designed to be recognized under motion blur.

Naimark and Foxlin [20] proposed a circular bar code called the Circular Data Matrix that is beneficial in terms of easy detection and ability to have a large number of uniquely identifiable codes. To address the issue of occlusion, Bergamasco et al. [4] proposed the RUNE-tag fiducial by creating a number of circular dots arranged in circular fashion. RUNE-tags can be detected even when 50% of the fiducial area is occluded. Bergamasco et al. [5] proposed the PiTag fiducial, also composed of circular structures but arranged in a rectangle, to exploit projective invariant cross-ratio. This provided similar occlusion resistance as RUNE-tag but with even less circular dots. *All of these techniques, however, rely on generic feature detection (e.g., circle detection) and such algorithms break down under motion blur.*

Tracking: Fiducial detection between successive video frames can be considered as a tracking problem where the tracked object is the fiducial. There is a very large body of research dedicated to tracking and interested readers are referred to [29] for a good survey.

Most tracking methods [24, 27, 23, 19] assume the image sequence to be blur free. In reality, however, the presence of motion blur in a video sequence is often unavoidable. To this end, Wu et al. [28] proposed the Blur-driven Tracker (BLUT) framework for tracking motion-blurred targets. BLUT is based on the observation that although motion blur degrades the visual features of the target, it may also provide useful cues about the movements to help tracking.

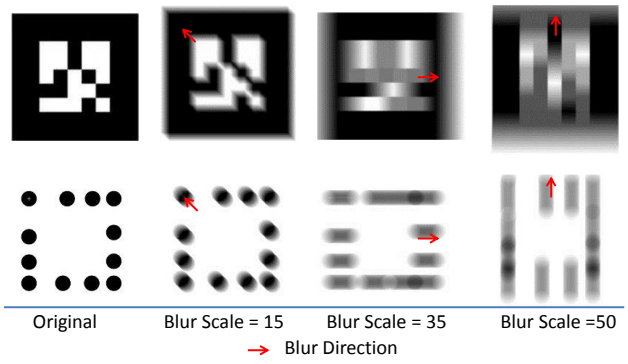
The BLUT framework successfully tracks a blurred target when there is uniform motion, and the position of the tracked object does not change drastically in successive frames. However, as we will demonstrate in Section 4, *the erratic motion from the quadcopters, as well as the problem of dropped video frames makes the task too problematic for BLUT to successfully track.*

3. Design of Blur Resistant Fiducial

We begin by first motivating the need for a new blur resistant fiducial by examining the performance of prior fiducials under motion blur. After this, we detail our design as well as the detection algorithm used to find the fiducial in an image, or a video sequence.

3.1. Prior Fiducials Under Motion Blur

Here we examine the performance of two popular fiducials under motion blur. Specifically we examine ARTags [13] and PiTags [5] given their differences in geometric design and the availability of an API to develop applications



Blur Scale	Recognition Rate	
	PiTag	ARTag
15	100	100
30	0	100
35	0	19
50	0	0

Figure 3: **Top:** The PiTag and ARTag fiducials blurred with various blur scales at different orientations. **Bottom Table:** Recognition rate (in percent). We see that the recognition rates for both the tags are significantly reduced. For severe blur, detection fails.

to recognize the tags. [1]

For controlled study, we simulate the appearance of the ARTag and PiTag markers by scaling the tags to 150×150 pixels. Both fiducials are then blurred using linear motion blur at various orientations with different blur scales. The blur motion ranged from 15 to 50 in magnitude (measured in pixels), representing small to significant motion blur. Figure 3 shows the visual appearance of the blurred tags. We then try to detect the markers using the ALVAR library [26] and PiTag library [22]. The table in Figure 3 shows the recognition rate (in percentage) of the two fiducials at various blur scales over all orientations. As we can see, the PiTag performance quickly diminishes under small amounts of blur, while at 35 units, the ARTag's recognition rate drops to less than 20%.

3.2. Blur Resistant Fiducial

We propose a binary coded fiducial that uses concentric white rings of equal widths on a black background with a blurred border[2]. The outermost and innermost rings represent the start and end of the code and is embedded in the fiducial. The binary code is represented by the presence (or absence) of rings between "marker" rings.

Depending on which ring is present or absent, the resulting binary code will change. The number of different patterns depends on the number of bits in the binary code.

[1] RUNE-tag [4] currently does not provide access to an implementation to generate or recognize its tags. The circular data matrix [20] is available as a commercial product, however it requires proprietary hardware.

[2] Obviously, this design can be inverted to have a white background with black rings.

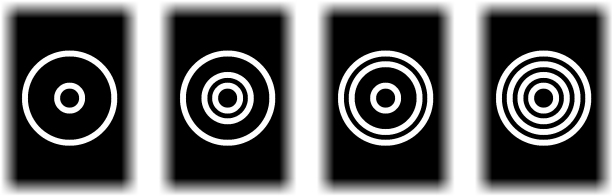


Figure 4: Two bit, binary fiducials, representing, from left to right 00, 01, 10, and 11.

For example, if the binary code has three bits, there will be a maximum of three rings between marker rings and we end up with eight different patterns. Figure 4 shows two bit binary fiducials.

Our fiducial detection strategy is different from [20, 5] and works under significant amounts of blur. As previously mentioned, our approach works under the observation that the motion blur for the quadcopter's camera can be well modeled as linear motion. This linear motion blur assumption has been shown to be reasonable in prior works [2, 3] targeting camera motion blur. Under this assumption, the scene content perpendicular to the blur direction is unaffected by the blur. Because of our circular design, the direction perpendicular to the linear motion will still be recognizable as a linear pattern. Figure 5 shows examples of this using a pattern under various motion directions.

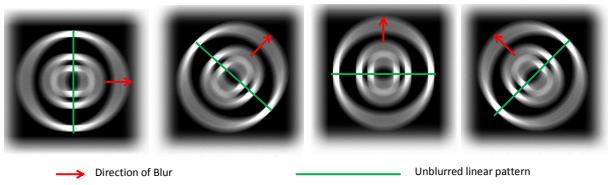


Figure 5: Change in blur direction changes the location and orientation of an unblurred linear pattern.

3.3. Detection Algorithm

Figure 6 shows the process involved in fiducial detection. We give a brief overview of our algorithm here with each step described in detail afterwards. Our detection algorithm has four steps. In Step 1, we apply a Gabor filter on the image to isolate the potential locations of the pattern. In Step 2, we find clusters of patches in the Gabor output. In Step 3, we perform Principal Component Analysis (PCA) on each cluster to find the dominant direction unaffected by blur. Finally in Step 4, based on the direction detected, we extract the intensity profile of the pattern and classify the fiducial.

Gabor filter: A 2D Gabor filter is a Gaussian kernel function modulated by a sinusoidal plane wave [17]. It is used to find high gradient patches. In our case, it is used to detect portions of the circular fiducials that were not af-

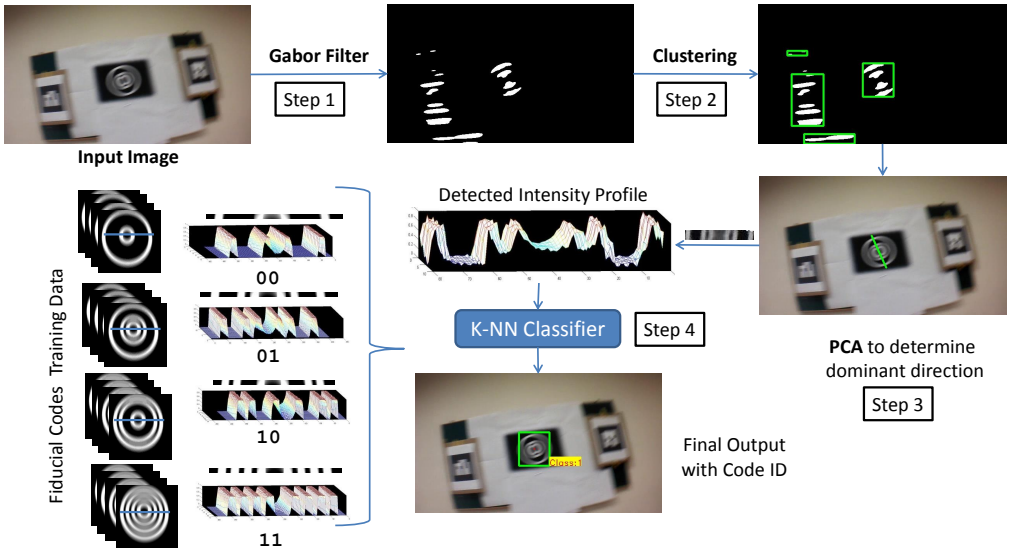


Figure 6: An overview of our algorithm. The four step process includes (Step 1) Filtering, (Step 2) Component clustering, (Step 3) Dominant direction determination and (Step 4) Classification using prior training data.

fected by the blur. We applied the Gabor filter in eight different orientations ($\theta = 0, 45, 90, \ldots, 225, 270, 315$). The following parameters were used for creating each Gabor kernel: λ (wavelength) $= 8$, γ (aspect ratio) $= 0.5$, σ (spread) $= 0.56\lambda$, ψ (phase angle) $= 0$ (for real part), $\pi/2$ (for imaginary part). Then ℓ^2-norm of outputs along all orientations is calculated and finally ℓ^2-normalized image is thresholded with threshold set to 0.4 (on scale of zero to one). For further details to the Gabor filter, see [17].

Clustering: The binarized Gabor filter responses are treated as a set of connected components in the image. Clustering is used to find components that are located in a close spatial region. We do this via hierarchical clustering [6] using unweighted average linkage with a distance threshold set to 150.

PCA: For each clustered set of components, we apply PCA to determine the dominant direction. This is done by examining the orientation of the first principal component. We then extract an intensity profile patch in the input image along this direction as it extends through the bounding box of the cluster. The signature of this profile will be used to identify the code.

After finding the intensity profile, we project the pixel intensities, and record the number and width of the white-to-black transitions. If the number of transitions, or the width of the transitions is not consistent with what is allowable by our code, we reject the clustered region as a potential fiducial. Clustered components that have allowable transitions counts and transitions with uniform widths are further considered for classification to determine the binary code.

Classification: As mentioned above, a small image patch containing the intensity profile of the fiducial is used to identify which of the many fiducials might be present and is represented by a code. We found that a training-based method using the k nearest neighbor (k-NN) technique gave better results than trying to find binary code directly by determining the presence (or absence) of ring at particular positions in the image patch. This required training-data which was easy to generate. A synthetic fiducial is blurred along 36 orientations (0, 10, 20, ..., 350) with blur scale set to 40, and the intensity profile along the first principal component from every output is taken as training data for that fiducial. Figure 7 shows the process of creating training data for fiducial with binary code "01" embedded in it. For a query image patch, we normalize the intensity range, and then compare this information against the training data. The class label from the closest top $K = 5$ images in the training data is used to label the patch.

4. Experimental Validation

We have implemented our algorithm in C++ using the OpenCV library. Experiments were performed on a laptop with Intel Core i7 processor(@3.4GHz) and 4GB RAM. (Please check author's webpage `http://www.comp.nus.edu.sg/~mgprasad/fiducial/` for details of source code and all datasets)

Our system has been tested on several image sequences captured from an AR Drone quadcopter. The drone was flown indoors looking at patterns attached to various walls. Each image sequence contains frames of different fiducials. Some sample outputs for each fiducial is shown in Figure 8. Our detection process takes around 0.3 seconds which trans-

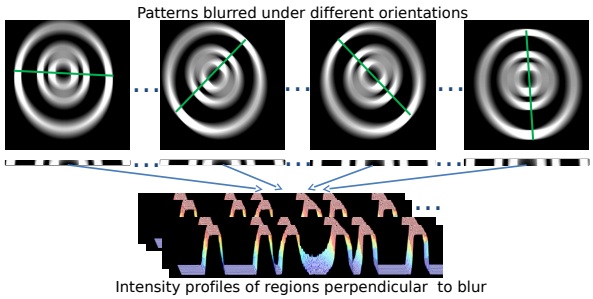


Figure 7: Creating training data for the fiducial with binary code "01": The synthetic pattern is blurred along various orientations and the intensity profile monitored and stored. The same process is used to create training data for all other fiducials.

lates to slightly over three frames per second.

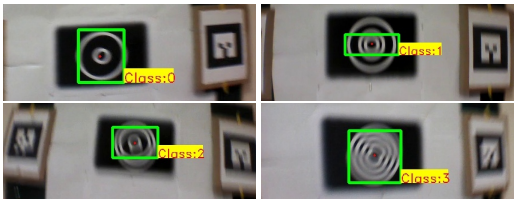


Figure 8: Output of our fiducial detection algorithm on sample images. The class label shown is the decimal equivalent of the binary coded fiducial.

Our system has also been tested on images containing multiple fiducial patterns in the same frame. Our algorithm successfully detected all fiducials as well as correctly classified them as shown in Figure 9.

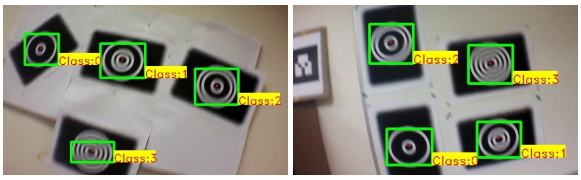


Figure 9: Multiple fiducials in the same frame can be efficiently detected.

We compare our results with the commonly used ARTag. We also compare our results with Blur-driven tracker (BLUT)[28].

4.1. Comparison with ARTag

First, we repeat the same blur simulation experiment (Section 3.1) on the proposed fiducial. Specifically, we build our blur resistant patterns at 150×150 pixel resolution and blurred them along various orientations with different blur scales.

Qualitatively, we see in Figure 10 that despite of even more blur than earlier described (50 units or more), detection is still feasible and practical. Quantitatively, the comparison of recognition rate is shown in Figure 11. Even

Figure 10: On synthetic experiments, we increase the blur much beyond the best possible results (see Figure 3) for prior methods. Even upto 65 units, the algorithm is successful.

under more severe blur, of 65 pixels, the algorithm is successful. Our recognition is 100% for all codes except the "00" which is undetectable after 50 pixels blur (which is still significantly better than ARTag). Reasons for the lower performance when"00" tags are present, is discussed in Section 5.

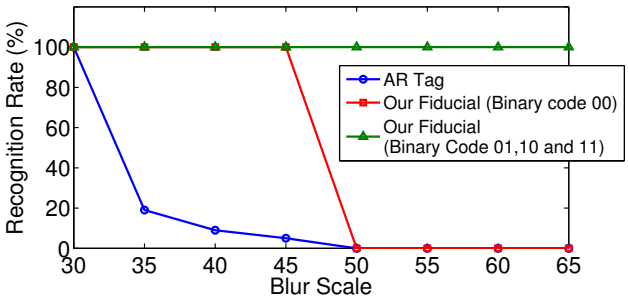


Figure 11: Comparison of recognition rate of fiducials. The proposed fiducial handsomely outperforms prior methods.

We also performed analysis to detect how accurately the center of the blur pattern can be localized under blur. To do this, we have simulated data along different blur orientations for all patterns over eight different blur scales (30 to 65 with step size of 5). Values for the localization of the pattern "00" after blur with 50 pixels is omitted since the pattern cannot be detected. From Table 1, it can be clearly seen that the mean error in locating center by our algorithm is within approximate 3 pixels, or 3% of the diameter of fiducial.

Blur angle	Mean(±std) error (in pixels)
0	1.33 ± 0.40
22	2.42 ± 1.05
45	1.46 ± 0.65
67	2.39 ± 1.28
90	1.83 ± 0.25

Table 1: Center localization error. Error is computed for various blur angles over various scales.

4.2. Real Data

We also compared results on the recorded feed using the AR Drone quadcopter. In our experimental setup, we have placed two ARTags in the scene along with our fiducial to compare the resilience of blur by each fiducial type. We

Test #	Number of frames	Binary Code	Recognition Rate (%)	
			ARTag	Our Fiducial
1	1205	00	65.6	86.5
2	1047	01	61.9	94.1
3	1102	10	62.4	92.74
4	1081	11	60.3	93.54

Table 2: Recognition rate of ARTag and proposed fiducials on real data captured through AR Drone. Each row shows analysis of a test dataset captured for our fiducial with different binary codes embedded in it. Each dataset has around 1000 frames captured representing roughly two minutes of video.

have used ar_track_alvar, ROS Wrapper for the ALVAR library [21], to detect the ARTags from the stream captured with the quadcopter camera. In each test dataset, we have used different two bit binary coded fiducial and recorded video of around two minute duration (i.e., around 1000 frames). The quadcopter was flown in a routine manner in the room with its camera facing the wall. The comparison of the recognition rate is shown in Table 2. Recognition rate of our fiducials ranges from 86.5% to 94.1%, while the ARTag is 60.3% to 65.6%. Classification accuracy of all fiducials (ARTag as well as ours) was approximately 100%, i.e., when tags were detected they were the correct fiducials and not other objects in the scene.

In another setup we arranged our fiducials on four sides of a box and revolved the quadcopter around the box (See [25]). Later, we repeated the process by replacing our fiducial by ARTags. Figure 12 shows some frames from this sequence as well as the performance of both fiducials. The overall detection rate of our tags is 90% while that of ARTags is only around 60%.

4.2.1 Comparison with BLUT

We also compare our approach with tracking designed for blurred input scenes. We have used four image sequences (consisting of around 1000 frames each). Each sequence contains different fiducials so that we are able to contrast the performance of BLUT [28] with the proposed method. From the images in the top rows of Figure 13 and Figure 14, we can see that BLUT is able to track the fiducial when the position of fiducial does not change too much in successive frames. Also, it can be seen that once BLUT loses track of the fiducial, it is not able to recover. Since our approach detects the code in each frame, large changes in the pattern's position is not an issue. Some of our detection results are also shown in Figure 13 and Figure 14.

We have also found that even if we reset the BLUT tracker after it loses track, the tracker will once again malfunction after around 100 frames (approximately within 6 seconds). When we checked the timestamp data from image header captured through the AR Drone, we found that, there was a difference of 0.14 seconds between two successive frames indicating the dropping of video frame (normal 30fps should have a gap of 0.033 seconds). Also, there were around 50 instances in 1000 frames where the timestamp difference between two successive frames was greater than 0.1 seconds. As such, it appears one of the main culprits causing the BLUT tracker to fail is the dropping of frames combined with the unstable motion of quadcopter, resulting in large discrepancies in the position of the fiducial between successive frames.

5. Discussion and Limitations

We have demonstrated the effectiveness of our blur invariant fiducial both on synthetic data, and on real video clips captured from a quadcopter. Our approach obtains a recognition rate of 86%–95% in real scenes compared to existing methods that average around 64%. We discuss some limitations of our approach in this section.

Our current processing time (0.3 seconds per frame) does not provide real-time performance. As such, we envision the method will be used in an offline manner for performing analysis of flight paths. Code profiling revealed that the Gabor filtering along the eight directions takes most of the time (0.03 – 0.04 seconds per orientation). Either an improved Gabor filter scheme is required or an alternative strategy for detection is required.

We found that sometimes our detection algorithm fails to recognize the "00" fiducial, when there is severe blur. This is because when there is too much blur, the innermost ring's response in the Gabor output is too low and not detected properly. This problem can be resolved by increasing the radius of innermost ring to reduce the effect of blur on the innermost ring.

We note that other markers are able to give full pose estimation after detection. However, we are only able to reliably detect the center point and therefore cannot estimate pose. Of course, if four markers were used in a known order, pose could be estimated. The current resolution of onboard camera is a significant hurdle to clear before we can effectively use multiple markers in each scene.

In terms of numbers of fiducials, we are able to generate less markers than ARTag. Many applications in robotics (e.g., quadcopter navigation) may not require simultaneous use of a large number of fiducials. Most of the time, it is sufficient to have 4-6 different fiducials. Nevertheless, we may be able to generate a larger number of fiducials by using color backgrounds. This is an area of interest for further research.

6. Conclusion

Quadcopters are subject to quick and unstable motions that can cause significant motion blur in the captured im-

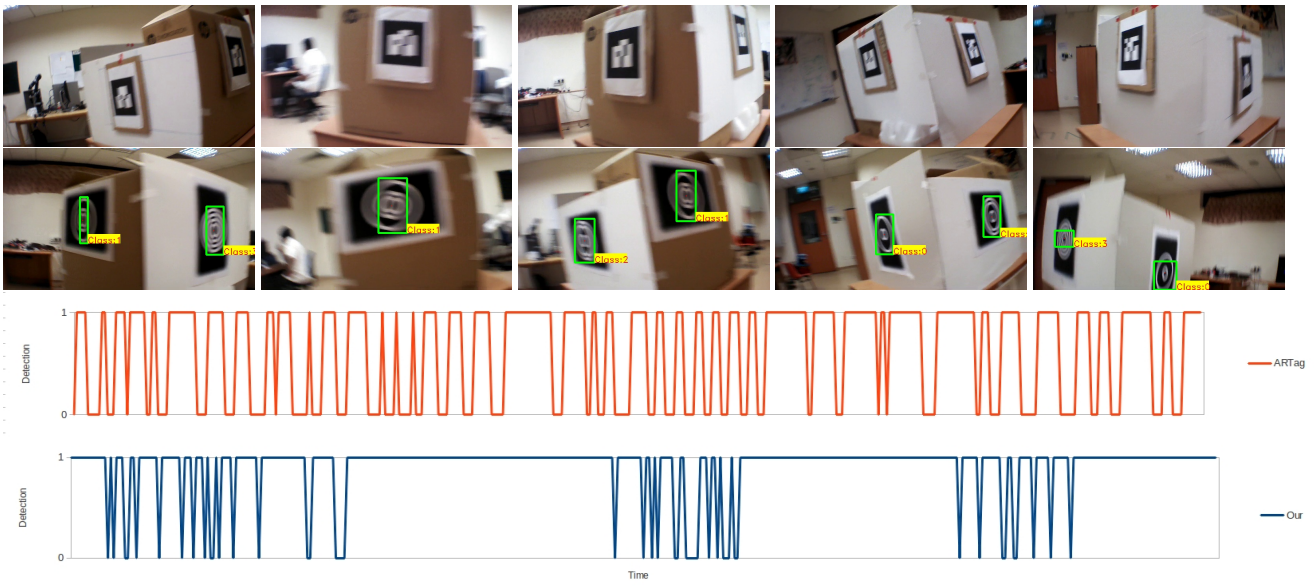


Figure 12: Comparison of ARTag and our fiducial when quadcopter is revolving around our setup. **Top** ARTags are not detected, shown by an absence of green rectangles. **Middle** In similar conditions, proposed fiducials are successfully detected. Overall detection rate of proposed fiducials is 90% while that of ARTags is around 60%. **Bottom Rows** Time view of detection. ARTag is "choppy" with frequent losses, while the proposed fiducial (extreme bottom) is "up" almost all the time.

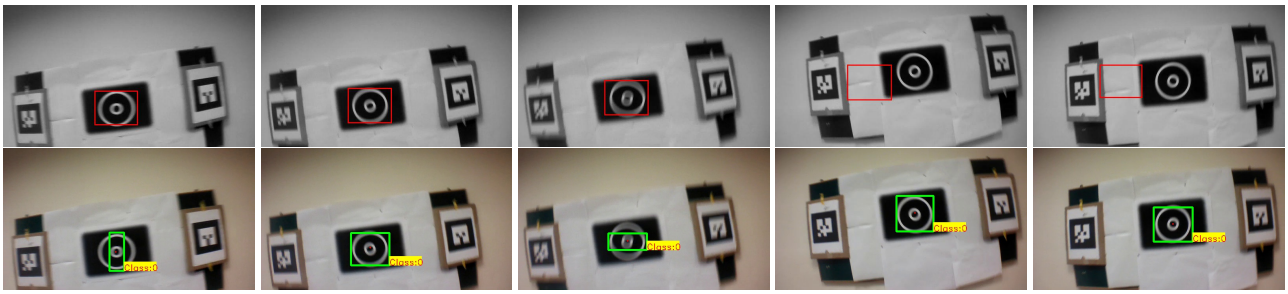


Figure 13: **Top:** Output of BLUT [28]. **Bottom:** Output of our algorithm on the same image sequence. BLUT is able to track the fiducial till the third frame, but from the fourth frame onwards, BLUT loses track. In the first three frames, the size of the bounding box is low, but in the fourth and fifth frame it is large, indicating a sudden forward movement of the quadcopter.

ages. This severely affects the detection rate of existing fiducials. We proposed the design of a fiducial that is resistant to motion blur. Our design of contrasting concentric rings is based on the observation that the direction perpendicular to the motion blur direction will be unaffected by the blur and therefore still be recognizable. We have shown through experimental validation that our fiducial will work under large amounts of motion blur and can significantly outperform existing fiducials under this scenario.

References

[1] ARToolkit. Last visited Nov 10, 2014. http://www.hitl.washington.edu/artoolkit. 1, 2

[2] M. Ben-Ezra and S. K. Nayar. Motion deblurring using hybrid imaging. In *CVPR*, 2003. 3

[3] M. Ben-Ezra and S. K. Nayar. Motion-based motion deblurring. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 26(6):689–698, 2004. 3

[4] F. Bergamasco, A. Albarelli, E. Rodolà, and A. Torsello. RUNE-Tag: A high accuracy fiducial marker with strong occlusion resilience. In *CVPR*, 2011. 1, 2, 3

[5] F. Bergamasco, A. Albarelli, and A. Torsello. Pi-Tag: a fast image-space marker design based on projective invariants. *Machine Vision and Applications*, 24(6):1295–1310, 2013. 1, 2, 3

[6] S. Bochkanov. ALGLIB. Last visited Nov 12, 2014. http://www.alglib.net. 4

[7] M. Bošnak, D. Matko, and S. Blažič. Quadrocopter hovering using position-estimation information from inertial sensors and a high-delay video system. *Journal of Intelligent and Robotic Systems*, 67(1):43–60, 2012. 1

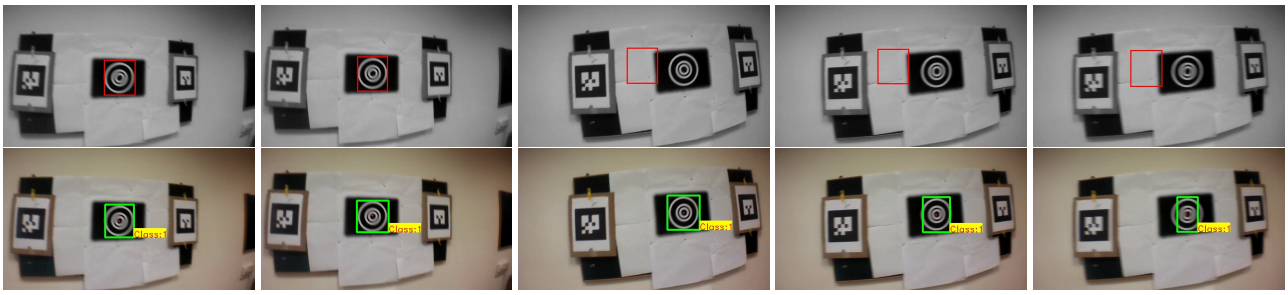


Figure 14: **Top:** Output of BLUT [28] on a sequence containing the "01" binary coded fiducial. **Bottom:** Output of our algorithm on the same image sequence. BLUT loses track from the third frame onwards.

[8] Y. Cho and U. Neumann. Multiring fiducial systems for scalable fiducial-tracking augmented reality. *Presence: Teleoperators and Virtual Environments*, 10(6):599–612, 2001. 2

[9] Y. Cho, J. Park, and U. Neumann. Fast color fiducial detection and dynamic workspace extension in video see-through self-tracking augmented reality. In *Fifth Pacific Conference on Computer Graphics and Applications*, 1997. 2

[10] A. J. Davison, I. D. Reid, N. D. Molton, and O. Stasse. Monoslam: Real-time single camera slam. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 29(6):1052–1067, 2007. 1

[11] J. Engel, J. Sturm, and D. Cremers. Camera-based navigation of a low-cost quadrocopter. In *International Conference on Intelligent Robots and Systems (IROS)*, 2012. 1

[12] J. Engel, J. Sturm, D. Cremers, and T. München. Semi-dense visual odometry for a monocular camera. In *ICCV*, 2013. 1

[13] M. Fiala. Artag, a fiducial marker system using digital techniques. In *CVPR*, 2005. 1, 2

[14] L. Gatrell, W. Hoff, and C. Sklair. Robust image features:concentric contrasting circles and their image extraction. In *Proc. of Cooperative Intelligent Robotics in Space*. SPIE, 1991. 2

[15] H. Kato and M. Billinghurst. Marker Tracking and HMD Calibration for a Video-based Augmented Reality Conferencing System. In *Proceedings of the 2nd International Workshop on Augmented Reality (IWAR)*, 1999. 2

[16] M. Klopschitz and D. Schmalstieg. Automatic reconstruction of wide-area fiducial marker models. In *International Symposium on Mixed and Augmented Reality (ISMAR)*, 2007. 1

[17] P. Kruizinga and N. Petkov. Non-linear operator for oriented texture. *IEEE Transactions on Image Processing*, 8(10):1395–1407, 1999. 3, 4

[18] H. Lim and Y. S. Lee. Real-time single camera slam using fiducial markers. In *ICROS-SICE International Joint Conference*, 2009. 1

[19] X. Mei and H. Ling. Robust visual tracking using l1 minimization. In *ICCV*, 2009. 2

[20] L. Naimark and E. Foxlin. Circular data matrix fiducial system and robust image processing for a wearable vision-inertial self-tracker. In *International Symposium on Mixed and Augmented Reality (ISMAR)*, 2002. 1, 2, 3

[21] S. Niekum. ROS Wrapper for ALVAR library. Last visited Nov 12, 2014. http://wiki.ros.org/ar_track_alvar. 6

[22] M. Nösner. ROS Wrapper for PiTag Detection Library. Last visited Nov 12, 2014. http://wiki.ros.org/cob_fiducials. 3

[23] P. Perez, C. Hue, J. Vermaak, and M. Gangnet. Color-based probabilistic tracking. In *ECCV*, 2002. 2

[24] D. Ross, J. Lim, R.-S. Lin, and M.-H. Yang. Incremental learning for robust visual tracking. *International Journal of Computer Vision*, 77(1-3):125–141, 2008. 2

[25] This paper video. Last visited Nov 10, 2014. http://www.youtube.com/watch?v=ibf9P-Coofw. 6

[26] VTT Technical Research Centre, Finland. ALVAR, an open source AR tag tracking library. Last visited Nov 12, 2014. http://virtual.vtt.fi/virtual/proj2/multimedia/alvar/. 2, 3

[27] Y. Wu, J. Cheng, J. Wang, and H. Lu. Real-time visual tracking via incremental covariance tensor learning. In *ICCV*, 2009. 2

[28] Y. Wu, H. Ling, J. Yu, F. Li, X. Mei, and E. Cheng. Blurred target tracking by blur-driven tracker. In *ICCV*, 2011. 2, 5, 6, 7, 8

[29] A. Yilmaz, O. Javed, and M. Shah. Object tracking: A survey. *ACM Computing Surveys*, 38(4), 2006. 2