2020 International Conference on Unmanned Aircraft Systems (ICUAS)
Athens, Greece. September 1-4, 2020

Experimental Comparison of Fiducial Markers for Pose Estimation

Michail Kalaitzakis*, Sabrina Carroll*, Anand Ambrosi*, Camden Whitehead*, Nikolaos Vitzilaios*

***Abstract*— Accurate localization is crucial for the autonomous navigation and control of Unmanned Aircraft Systems (UAS). In most applications, localization is provided from a Global Navigation Satellite System like GPS and Galileo or more recently from Visual Odometry, Visual-Inertial Odometry, and Simultaneous Localization and Mapping methods. In many cases though, especially when precise maneuvers are required, fiducial markers are used. Fiducial markers are able to provide accurate localization data and have been used in many applications where reliable pose measurements are needed for specific objects or locations. This paper presents an experimental comparison of four different open-source fiducial markers that are widely used in UAS applications (ARTag, AprilTag, ArUco, and STag). The fiducial markers are evaluated based on their localization capabilities as well as their computational efficiency. To facilitate the comparison, a ROS package for the STag marker is developed and publicly released.**

I. Introduction

In the recent years, Unmanned Aircraft Systems (UAS) have become widely used in aerial photography, mapping, surveying and remote inspections [1]–[3]. Autonomous UAS rely on accurate localization schemes to ensure their safe navigation. In many cases, Global Navigation Satellite Systems (GNSS) are used to provide the location of the UAS. The recent developments in Visual Odometry (VO) and Simultaneous Localization and Mapping (SLAM) have made possible the deployment of UAS in GNSS-degraded and GNSS-denied environments as well.

While VO and SLAM methods are able to provide accurate localization data in structured environments with strong visual features, they fail to do so in environments without strong features and challenging lighting conditions [4], [5]. In such environments, to increase the localization accuracy, fiducial markers are often utilized to augment the localization capabilities of the platform. Fiducial markers offer a highly distinguishable pattern with strong visual characteristics that also feature specific encoding as a fail-safe against misdetections. Other uses of fiducial markers in UAS and Robotics research include: to pinpoint specific objects such as landing platforms and payloads [6], [7], to mark multiple platforms in robotic teams [8], and to mark objects that a robot needs to handle or use [9].

There are many different fiducial marker packages available, including some which are offered as open-source. A brief summary of the tags available is included in [10]. Many of these open-source packages are also offered to the Robotics community as ROS packages [11]. Although there are some comparisons between the different packages, most focus on the library size offered and the detection reliability. To the knowledge of the authors, the only comparison paper that focuses on the localization performance and computational cost for each package is [12], where three packages are evaluated for their performance in the underwater environment domain.

The aim of this work is to offer the research community an experimental comparison of four different fiducial marker packages: ARTag [13], [14], AprilTag [15], [16], ArUco [17], and STag [10]. The selection of these particular markers is based on the fact that: (i) they are state-of-the-art markers widely used in latest UAS applications and (ii) they are open-source. Moreover, the first three offer a ROS implementation, while for the purposes of this work, we have developed and made publicly available a ROS implementation of STag as well. It is of note that for this paper we are using the ROS implementations for all the algorithms "out-of-the-box" without making any hardware specific optimizations like in [6]. Specifically, the contributions of this paper are:

- The experimental comparison of four fiducial marker packages on detection rate and localization accuracy from varying distances and orientations of the markers.
- Computational performance evaluation for three different computer systems widely used in autonomous platforms.
- The development and public release of a ROS package for the STag fiducial marker.

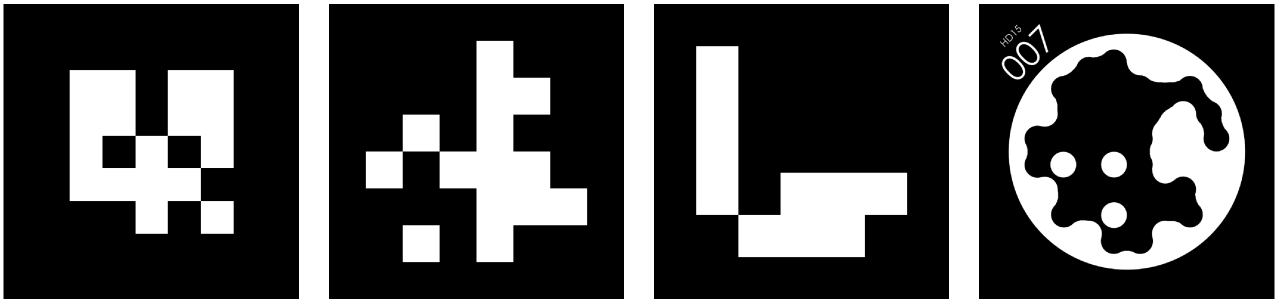
The paper is organized as follows: Section II offers a background on fiducial markers and presents a few notable examples of use cases on UAS research. Section III details the experimental setup and outlines the different experiments performed for the marker comparison. Section IV offers a presentation and discussion of the experimental results. Finally, Section V draws the conclusions of this work.

II. Background

Fiducial markers in their general form are objects used to provide a point of reference or a measurement in an image. Applications range across various disciplines from medical imaging to PCB manufacturing. The fiducial markers used in UAS and Robotics applications were initially designed for augmented reality applications. They are artificial landmarks of known size and shape that feature a specific pattern that is used to identify them. While in most cases the markers are black and white, there are some packages that use colored markers [18]. Although colored markers show advantages like decreased detection time and false-positives, they are not robust to increased distances and steep angles.

*M. Kalaitzakis, S. Carroll, A. Ambrosi, C. Whitehead and N. Vitzilaios are with the Department of Mechanical Engineering, University of South Carolina, Columbia, SC, USA {michailk, src2, aambrosi, camdencw}@email.sc.edu, vitzilaios@sc.edu

978-1-7281-4277-7/20/$31.00 ©2020 IEEE



(a) (b) (c) (d)

Fig. 1: The markers used in this work: (a) ARTag [13], [14], (b) AprilTag [15], [16], (c) ArUco [17], and (d) STag [10].

Most square-shaped markers are based on ARToolkit [19], a marker tracking system initially created for video-based augmented reality conferencing. To detect the marker positions, a global threshold is used to identify regions whose outline can be fitted by four line segments. The internal patterns used on ARToolkit markers can be of any shape, and image correlation is used to identify the different markers. This enables the use of markers that are easily distinguishable by the users but the underdefined nature of the markers leads to an increased number of false detections and misdetections.

A. *Fiducial markers used in this work*

The four fiducial markers being evaluated in this work are shown in Figure 1.

ARTag [13], [14], is based on ARToolkit but it uses digital coding theory to create the marker's internal pattern. By making the interior of the square a six by six bit matrix, each marker is given a unique 36-bit long word as its ID. The marker library is created in order to ensure a minimum Hamming distance between the marker codewords. Furthermore, a gradient-based method is used to detect lines that are later grouped into quadrilaterals of candidate markers. These improvements led to increased tag detection reliability and enabled detections under partial occlusions.

AprilTag [15], [16], builds on the framework established by ARTag while offering a number of improvements. These include a graph-based image segmentation algorithm that analyzes gradient patterns on the image to precisely estimate lines and a quad extraction method that allows edges that do not intersect to be identified as possible candidates. A new coding system is also implemented to address issues stemming from the 2D bar-code system. These issues include incompatibility with rotation and false positives in outdoor conditions. As a result, AprilTag has an increased robustness against occlusions and warping as well as a decreased amount of misdetections.

ArUco [17], is another package based on ARTag and ARToolkit. The most notable contribution of ArUco is that it allows the user to create configurable libraries. Instead of including all possible markers in the standard library, users can elect to generate a library based on their specific needs. The library will only contain the specified number of markers with the greatest Hamming distance. An additional advantage of the smaller size of the custom libraries is the reduced computing time. For the purposes of this paper, the full library of 1,024 available tags was used.

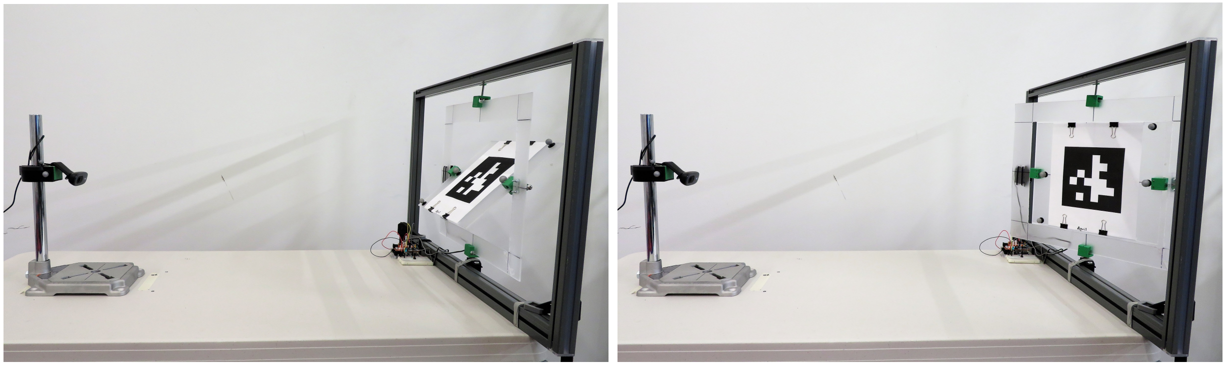
STag [10], is a newly developed fiducial marker package that focuses on the stability of the pose estimation measurements. Since consistent and stable measurements are very important in control applications, we decided to include this package to our comparison and to create a ROS package that is made available to the community. This is the first STag comparison to be performed outside of comparison tests performed by the developer. The major difference between STag and the rest of the packages used in this paper is that the markers used in STag have a circular pattern in their center. After the line segmentation and quad detection, an initial homography is calculated for the detected markers. Then, by detecting the circular pattern in the center of the marker, the initial homography can be refined using elliptical fitting. Elliptical fitting is shown to provide a better localization compared to the one provided using quads. This refinement step provides increased stability to the measurements.

B. *Applications*

Within the field of UAS and Robotics research there have been many cases where fiducial markers are utilized. In [20], [21], ArUco markers are used for the localization of low-cost robots in indoor and GNSS-restricted environments. Utilizing the positioning from the markers, the platforms are able to navigate their environments and follow specific trajectories. In [22], an approach to SLAM is presented that uses ARTags.

Another use of fiducial markers is for object identification and tracking or payload manipulation. In [23], a UAS is able to detect and follow an Unmanned Ground Vehicle (UGV) through cluttered environments that is equipped with a marker. The UAS uses the marker placed on the ground vehicle to identify and track it. In [7], a UAS uses a fiducial marker on a slung load to track its position while it carries it through different trajectories.

The most notable use of fiducial markers in the field of UAS is to mark a landing platform for the task of autonomous landing, especially in the case of a moving landing platform. In [24], [25] a UAS is able to land on a moving UGV by tracking a bundle of markers placed on the UGV. In both of these cases, different sized markers are used



(a) (b)

Fig. 2: Overview of the experimental setup with the two axes gimbal on the right side of each image, where (a) the marker position is altered in the longitudinal direction and (b) the marker position is altered in the lateral direction.

to allow the detection and tracking of the platform during the landing process. In [26], a fiducial marker is used on an inclined platform to calculate the relative position and orientation of the platform in order to land on it. In [27] a UAS is tracking a set of two different marker bundles placed on an Autonomous Surface Vehicle (ASV) to mark the landing platform. The UAS is able to repeatedly take-off and land from the ASV making possible long-duration joint missions for the two platforms. Other notable examples are [6], [28] where a UAS autonomously lands on a high velocity ground vehicle at speeds of up to $30km/h$ and $50km/h$ respectively.

III. Experimental Setup

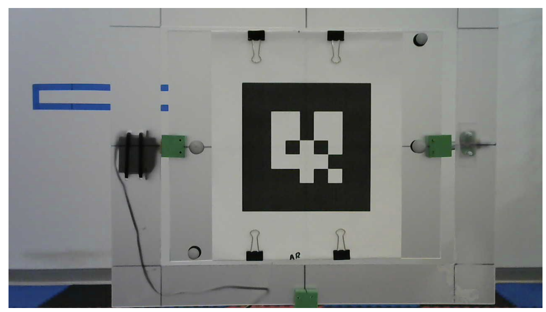
For the purposes of this comparison, we are evaluating the efficiency of the four packages on three different small factor computers that are commonly used in UAS applications. The computers used are the Raspberry Pi 3B+, the NVidia Jetson TX2, and the Intel NUC (NUC5i7RYH). The Pi is a popular low-cost solution, while the TX2 and the Intel NUC are selected for their powerful GPU and CPU, respectively. The specifications for each computer are shown in Table I.

TABLE I

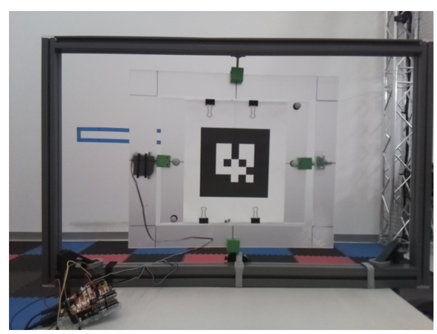
	Raspberry Pi 3b+	Jetson TX2	Intel NUC
Total Memory	1GB	8GB	16GB
Type of RAM	LPDDR2 SDRAM	LPDDR4	DDR3L
CPU Cores	4	2 + 4	2
CPU - Hz Rate	1.4GHz	2GHz + 2GHz	3.1 GHz
GPU Cores	None	256	48
GPU - Hz Rate	N/A	1.3GHz	0.3GHz

Specifications for each computer system used.

As for camera sensors, a Logitech C270 Webcam and a Raspberry Pi camera version 2 module (piCam) were used. The Logitech was used to capture a high resolution video of 1280x720 pixels at 25 fps while the piCam was used to collect data at a resolution of 640x480 pixels at 20fps. We chose to use a piCam for the low resolution images since it is commonly used with the Pi. An image from both cameras can be seen in Figure 3 at a distance of $0.75m$ from the marker. It is clear that the piCam has a larger field of view and a higher contrast than the Logitech camera.



(a)



(b)

Fig. 3: Images from (a) the Logitech camera and (b) the Raspberry Pi camera at a distance of $0.75m$ from the marker.

Figure 2 shows an overview of the experimental setup. The different markers are clipped on a marker base and a gimbal has been developed to actuate this base in two perpendicular axes (longitudinal and lateral). The cameras are mounted on a stand that has adjustable height and can be moved to different distances from the marker base. Both the camera stand and the marker base are tracked using an OptiTrack Motion Capture (MoCap) system to provide an accurate relative position and orientation between them. This ensures that an accurate ground truth is available even in cases where there are small changes in the setup between different experiments. Along with the videos from the cameras, the pose data from the motion capture system are streamed and saved at a rate of 100Hz into ROS bag files so that the same data can be processed on all computers.

To investigate the detection rate and accuracy of the package associated with each marker, data are collected at various different stationary distances and angles. The distances considered are $0.75m$, $1.0m$, $1.5m$, $1.75m$, and $2.0m$ at an angle perpendicular to the camera. Two different lighting conditions are considered as shown in Figure 4 to determine if the lighting conditions affect the detection rate and the accuracy of each package. A 60 second video is captured for each package at each distance resulting in a total of 1500 images for the high resolution camera and 1200 images for the piCam.

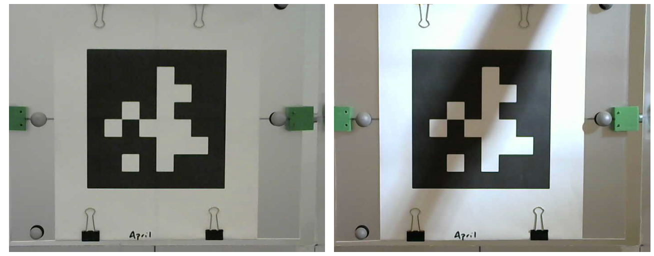
To test the performance at various angles, the gimbal rotates in $10°$ increments up to $80°$, along the longitudinal axis, with the camera placed $1.0m$ away from the marker. The gimbal maintains the orientation of the marker for 15 seconds for each increment, resulting in 375 high resolution images and 300 low resolution images per angle. This test is only performed in the longitudinal direction because the lateral direction is expected to perform in the same manner. For these experiments two different lighting conditions are used, as in the distance tests described earlier, as shown in Figure 4. In this experiment we test how the angle of the camera affects the orientation and position measurement accuracy as well as the detection rate.

The next parameter assessed is the computational efficiency. Using the apparatus shown in Figure 2, a series of angle sweeps are completed along the two axes perpendicular to the camera direction. In the first motion, the longitudinal axis oscillates between ± 45 degrees while the lateral axis is stationary. Then the lateral axis oscillates between ± 45 degrees. In the final motion, both axes are varied between ± 30 degrees simultaneously to provide a variety of intermediate angles. The total duration of the sweep motion is 50 seconds and the experiment is repeated 3 times for each package.

To process the experimental data, the ROS packages for ARTag[1], AprilTag[2] and ArUco[3] are used on all three computers with the same configuration for each camera. Since a ROS package for the STag was not available prior to this work, we developed a ROS package that is now publicly released to the research community[4].

It is worth noting that there is a difference between the STag implementation described in [10] and the ROS implementation developed for this paper. While in the original paper the pose of the marker is found using the method described in [29], in the ROS implementation we are using OpenCV's Perspective-n-Point algorithm (solvePnP). While this may lead to decreased stability and accuracy in the case of a single marker, the algorithm described in [29] is specifically designed for coplanar points and planar targets. OpenCV's solvePnP on the other hand, can be used for any target thus facilitating bundles of non-coplanar markers. Both ArUco and AprilTag use solvePnP for bundle tracking.

[1]https://github.com/ros-perception/ar_track_alvar
[2]https://github.com/AprilRobotics/apriltag_ros
[3]https://github.com/pal-robotics/aruco_ros
[4]https://github.com/usrl-uofsc/stag



(a) (b)

Fig. 4: The AprilTag marker used (a) under normal lighting conditions and (b) with a shadow being cast as seen from the Logitech camera.

IV. RESULTS

A. Performance with varying marker distance

The accuracy of the pose measurement and rate of detection for each package at varying distances are summarized in Table II. The metrics used for the accuracy of pose measurements are the average error and the standard deviation of the error for all successfully detected images in the experiment. The average position and orientation are calculated in each setup and used to compare the measurements from the markers to make sure that any outliers in the ground truth will not affect the measured accuracy.

The detection rate shows the percentage of the images in which the correct marker was successfully detected. Successful detections include only the images where the marker with the correct ID was detected. In cases where multiple markers are falsely detected in an image, the successful detections include the images that include the correct marker.

Specifically, Table II, shows how the distance between the camera and the marker affect the measurement accuracy and the detection rate for each marker. The package with the highest detection rate is denoted in green, while the lowest error is noted in blue and the lowest standard deviation is noted in magenta. For the high resolution images (Logitech camera), AprilTag and ArUco show a consistent degrading in accuracy. Moreover, ARTag is the only package that is affected by the presence of shadows in the image at distances larger than $1.75m$. STag has a consistent performance at all distances. The results from the low resolution images (Pi camera) present maximum error at a distance of $2.0m$. AprilTag and ArUco are again the packages that present a consistent degradation. In general, the results from the low resolution images have a lower error in most settings. This may be due to the lower contrast of the Logitech camera that causes smoother transitions on edges and thus lower gradients making the corner detection less robust.

TABLE II

Camera	Distance (cm)	Light	ARTag			AprilTag			ArUco			STag		
			mean	std	rate	mean	std	rate	mean	std	rate	mean	std	rate
Logitech	75	normal	2.262	0.046	42.899	3.850	**0.017**	96.914	2.067	0.021	**97.072**	**1.863**	0.024	93.577
		shadow	2.204	0.024	43.130	3.900	0.021	**96.997**	2.034	0.027	95.976	**1.859**	**0.017**	93.090
	100	normal	2.183	**0.017**	43.699	4.721	0.021	**96.402**	2.204	0.028	95.926	**1.671**	0.067	89.325
		shadow	2.224	0.051	44.260	4.723	0.021	**96.163**	2.184	0.028	96.030	**1.716**	**0.007**	91.958
	150	normal	2.484	0.442	43.375	5.981	**0.073**	**96.644**	2.242	0.112	96.165	**1.450**	0.249	95.562
		shadow	1.909	0.212	43.634	6.052	**0.106**	**97.068**	2.443	0.315	96.071	**1.425**	0.108	95.007
	175	normal	2.668	0.344	43.546	6.559	0.146	96.074	2.315	0.229	96.103	**1.012**	**0.107**	**96.112**
		shadow	ND	ND	0.0	6.639	0.121	**96.554**	2.620	0.197	95.994	**1.131**	**0.059**	96.036
	200	normal	3.155	0.220	43.611	7.236	**0.191**	96.012	3.050	0.315	96.083	**0.648**	0.257	**96.511**
		shadow	ND	ND	0.0	7.273	0.251	**96.473**	3.615	0.200	**96.473**	**0.680**	**0.192**	96.054
Pi camera	75	normal	1.431	**0.007**	47.544	2.142	0.012	**94.704**	1.284	0.011	94.693	**0.737**	0.077	90.606
		shadow	1.549	**0.015**	47.118	2.306	0.031	94.444	1.353	0.020	**94.829**	**0.764**	0.057	94.636
	100	normal	1.954	0.084	49.766	2.541	0.031	94.429	1.562	0.026	94.595	**0.842**	**0.023**	**98.448**
		shadow	2.214	0.053	47.349	2.785	0.042	**95.333**	1.813	0.100	93.417	**0.993**	**0.024**	**95.333**
	150	normal	2.617	0.280	47.313	3.292	0.232	**95.333**	3.632	**0.047**	94.051	**0.475**	0.757	75.805
		shadow	3.184	0.226	47.447	3.596	0.082	**94.865**	3.748	**0.078**	94.123	**0.495**	0.588	94.624
	175	normal	1.768	**0.192**	47.292	3.410	0.340	94.787	3.827	0.612	**94.917**	**0.093**	1.041	93.279
		shadow	3.824	**0.071**	46.891	4.081	0.235	**94.366**	4.838	0.696	93.001	**0.398**	0.540	93.361
	200	normal	2.697	0.289	47.627	3.510	0.212	94.429	3.780	1.361	95.067	**1.542**	**0.127**	**95.410**
		shadow	4.403	0.441	47.024	4.877	0.290	94.648	6.151	0.875	**95.087**	**1.194**	**0.240**	91.589

Distance Data - Distance Results: This table presents the calculated distance estimates based on the data collected when the position of the camera is moved from $75cm$ to $200cm$ from the marker while keeping the camera perpendicular. Distance, mean, and standard deviation (std) columns are all measured in centimeters and the rate is measured in percentages. The package with the greatest detection rate is denoted in green, while the lowest error is noted in blue and the lowest standard deviation is noted in magenta. The values marked with "ND", indicate that the tag was not detected in any of the frames.

The package with the best detection rate is AprilTag, while ARTag has a consistently low detection rate. AprilTag, ArUco, and STag have a better detection rate in the high resolution images and ARTag has a better performance in the low resolution images. Again, this may be explained by the lower contrast of the Logitech camera.

In terms of precision, STag presents the best results in all cases. As for the stability of the measurements, while STag is the package that has the minimum standard deviation in most settings, ARTag and AprilTag have comparable performances. In all the packages the standard deviation increases with the distance.

B. Performance with varying marker orientation

Tables III and IV show how the orientation of the marker affects the measurement accuracy and the detection rate. In particular, Table III shows how the orientation measurements and the detection rates are affected while Table IV shows the distance error at different marker orientations. In Table III, the package with the greatest detection rate for each angle and lighting condition is denoted in green, while the lowest error is noted in blue and the lowest standard deviation is noted in magenta. In Table IV, the lowest error is noted in blue and the lowest standard deviation is noted in magenta.

Table III shows that AprilTag has an overall better performance, especially based on the data from the Logitech camera. Although ArUco is consistently competitive in its detection rate, it generally has higher mean and standard deviation errors indicating lower accuracy than AprilTag. It can also be noted that the AprilTag, ArUco, and STag markers have detection rates greater than 90% except for one outlier in the STag marker at an angle of $80°$ under normal conditions on the low resolution data. This indicates a high aptitude to detect markers within images regardless of the angle or the abnormal lighting conditions considered in this experiment. ARTag consistently has a much lower detection rate hovering around 45% for the Logitech camera data and 49% for the piCam data. Despite the great disparity in the ARTag detection rate, the mean error and standard deviation are generally competitive with the other three platforms.

Overall, in the high resolution images, all packages tend to perform best at an angle between $40°$ and $70°$ where the estimated position is accurate within $1°$. This is also the range that consistently experiences some of the lowest standard deviations for each marker. Again, it is of note that the data collected using the piCam is more accurate (with lower mean error and standard deviation) than the data collected with the Logitech camera.

Table IV indicates that the distance estimations are more accurate when the marker is perpendicular to the camera. All packages experience a degradation in their accuracy as the angle between the camera and the marker increases with slight improvements at $70°$ and $80°$. For most angle offsets, STag presents the most accurate distance estimate, though in

TABLE III

Camera	Angle	Light	ARTag mean	ARTag std	ARTag rate	AprilTag mean	AprilTag std	AprilTag rate	ArUco mean	ArUco std	ArUco rate	STag mean	STag std	STag rate
Logitech	0°	normal	2.831	0.158	45.833	2.450	**0.068**	99.539	**2.386**	0.176	**99.552**	3.317	0.347	92.825
		shadow	3.132	0.102	45.045	3.417	**0.039**	**100.000**	**2.012**	0.053	99.545	3.380	0.092	94.444
	10°	normal	**0.625**	**0.047**	46.083	0.884	0.168	**99.552**	1.773	0.115	99.548	1.118	0.135	91.855
		shadow	2.039	**0.081**	45.455	**1.386**	0.136	**99.541**	1.401	0.131	99.539	1.421	0.329	93.953
	20°	normal	2.326	**0.055**	46.296	**1.517**	0.080	99.545	2.219	0.178	**100.000**	2.319	0.115	92.793
		shadow	3.129	0.089	45.045	**2.299**	**0.068**	**99.545**	2.744	0.074	99.543	2.971	0.089	93.088
	30°	normal	**1.162**	0.068	46.330	1.325	**0.066**	99.541	1.379	0.122	**100.00**	1.565	0.077	91.818
		shadow	1.527	0.125	45.662	**1.310**	**0.057**	99.537	1.455	0.105	**99.539**	1.728	0.302	94.907
	40°	normal	0.214	0.050	46.083	0.641	**0.024**	99.091	**0.203**	0.025	**99.550**	0.590	0.070	92.991
		shadow	0.564	0.063	45.662	**0.456**	**0.031**	99.087	0.910	0.050	**99.548**	0.605	0.186	96.789
	50°	normal	**0.141**	0.022	44.595	0.365	**0.015**	**99.543**	0.459	0.046	99.087	0.244	0.033	94.064
		shadow	0.156	**0.021**	45.662	**0.022**	0.024	99.528	0.162	0.046	**99.537**	0.047	0.204	95.909
	60°	normal	0.346	0.030	44.395	**0.126**	**0.009**	99.543	0.239	0.011	**100.00**	0.422	0.019	97.727
		shadow	0.473	0.020	45.455	0.264	**0.016**	**100.00**	**0.188**	0.018	**100.00**	0.194	0.558	95.045
	70°	normal	0.738	0.039	45.662	**0.476**	0.044	99.083	0.779	**0.015**	**99.545**	0.699	0.064	99.541
		shadow	0.716	**0.009**	45.872	0.399	0.012	**99.103**	0.607	0.013	99.087	**0.097**	0.282	96.330
	80°	normal	1.270	**0.039**	41.176	**1.052**	0.089	**100.00**	1.238	0.046	99.087	1.258	0.042	98.190
		shadow	1.143	0.042	45.045	0.874	**0.009**	99.552	0.976	0.010	**100.00**	**0.602**	0.235	97.273
Pi camera	0°	normal	1.883	**0.110**	50.249	0.649	0.267	99.010	**0.602**	0.604	99.010	0.627	0.363	**99.502**
		shadow	**0.010**	**0.147**	49.751	0.288	0.611	**99.502**	4.863	0.435	**99.502**	0.411	0.419	**99.502**
	10°	normal	1.532	0.200	49.751	**0.476**	**0.092**	97.887	1.024	0.178	**99.502**	0.571	1.902	**99.502**
		shadow	1.521	0.100	49.451	**0.323**	0.222	**99.502**	4.201	**0.041**	**99.502**	1.801	0.874	**99.502**
	20°	normal	0.062	**0.052**	49.254	**0.028**	0.054	**99.502**	0.138	0.080	**99.502**	0.088	0.168	**99.502**
		shadow	0.487	**0.063**	49.751	0.561	0.152	**99.502**	**0.278**	0.111	**99.502**	0.600	0.184	98.701
	30°	normal	0.256	**0.039**	49.367	**0.035**	0.055	99.502	0.765	0.141	99.502	0.479	0.158	**100.000**
		shadow	0.294	**0.036**	49.751	0.288	**0.036**	**99.502**	0.255	0.067	**99.502**	**0.134**	0.097	**99.502**
	40°	normal	0.092	0.038	49.254	**0.034**	**0.016**	98.658	0.296	0.031	99.502	0.457	0.387	**100.000**
		shadow	0.148	0.034	49.751	**0.026**	0.075	99.005	0.502	**0.016**	**99.502**	0.497	0.025	**99.502**
	50°	normal	0.440	0.059	48.795	0.519	0.025	**99.502**	0.134	**0.018**	**99.502**	**0.080**	0.032	**99.502**
		shadow	0.210	0.018	49.751	**0.025**	0.020	99.502	0.207	**0.016**	99.502	0.434	0.020	**100.000**
	60°	normal	0.163	**0.015**	49.254	**0.022**	0.016	97.887	0.528	0.033	**99.502**	0.026	0.048	**99.502**
		shadow	0.477	0.015	49.412	**0.066**	0.019	**99.502**	0.193	**0.010**	99.005	0.172	0.034	97.279
	70°	normal	**0.131**	0.015	49.751	0.137	0.011	97.902	0.299	**0.007**	**100.000**	0.322	0.048	98.026
		shadow	**0.071**	0.031	49.751	0.175	0.031	**99.502**	0.342	**0.009**	99.005	0.184	0.152	98.990
	80°	normal	0.335	0.019	49.020	**0.235**	0.012	**99.502**	0.518	**0.004**	**99.502**	0.277	0.169	36.634
		shadow	0.169	0.097	49.505	**0.010**	0.025	**100.00**	0.153	**0.016**	99.502	0.257	0.073	94.527

Angle Data - Angle Results: The table presents the calculated angle orientation of the marker when the camera is kept at a distance of $1m$ from the target and the angle is varied from $0°$ to $80°$. Angle, mean, and standard deviation (std) are all measured in degrees and the rate is measured in percentages. The package with the greatest detection rate is denoted in green, while the lowest error is noted in blue and the lowest standard deviation is noted in magenta.

rare cases ArUco or ARTag outperform it. However, STag also has the highest standard deviation in nearly all cases indicating a low stability. In comparison, ARTag or AprilTag feature the lowest standard deviations with regards to the distance estimation for the varied angles.

C. Computational Cost

Figures 5, 6, and 7 show box plots for the computational cost of each package on the NUC, TX2, and Pi platforms, respectively. The CPU usage is reported as the percentage of a single CPU core of the platform while the memory usage is shown as the percentage of the total memory used. While the performance of both the high and low resolution images are recorded on the NUC and TX2 platforms, only the low resolution images are used in the Raspberry Pi. When the high resolution data was attempted on the Pi, neither AprilTag nor STag could process the images fast enough to provide a detection rate of more than 1Hz, making them impractical for use on the Pi. ARTag was able to provide data at a rate of 10Hz with ArUco reaching a rate of 9Hz.

In all platforms and in both resolutions, AprilTag is the most computationally expensive package followed by STag. ARTag is the least expensive while ArUco has comparably low computational needs. Specifically, AprilTag uses a complete core or more on the NUC and close to two cores on both the TX2 and the Pi. This translates to half the processing power for both the NUC and the Pi. ARTag, ArUco, and STag all require significantly less resources to compute than AprilTag except in the case of STag on the high resolution images on the NUC.

As for memory requirements, AprilTag requires the most resources followed by ARTag with both STag and ArUco presenting low memory needs. Contrary to the CPU usage though, the memory usage for all packages is insignificant, reaching a maximum of 11% across all platforms.

TABLE IV

Camera	Angle	Light	AR mean	AR std	April mean	April std	ArUco mean	ArUco std	STag mean	STag std
Logitech	0°	normal	2.370	0.052	5.097	**0.029**	2.294	0.045	**1.435**	0.138
		shadow	2.260	**0.027**	4.782	0.033	2.246	0.031	**1.895**	0.074
	10°	normal	3.623	**0.019**	6.318	0.045	3.803	0.021	**3.004**	0.056
		shadow	3.741	**0.033**	6.295	0.035	**3.373**	0.040	3.514	0.128
	20°	normal	5.512	**0.012**	7.934	0.013	5.585	0.021	**4.854**	0.032
		shadow	5.487	**0.009**	7.673	0.011	**5.067**	0.017	5.199	0.018
	30°	normal	6.536	**0.013**	8.878	0.016	6.623	0.015	**5.697**	**0.013**
		shadow	6.498	**0.008**	8.627	0.010	6.427	0.010	**6.169**	0.027
	40°	normal	7.016	**0.008**	9.346	0.010	7.232	0.011	**6.239**	0.013
		shadow	6.983	**0.006**	9.156	0.007	6.905	0.009	**6.755**	0.032
	50°	normal	7.167	0.011	9.463	0.009	7.391	**0.008**	**6.328**	0.010
		shadow	7.082	**0.005**	9.262	0.008	7.035	0.009	**6.746**	0.026
	60°	normal	6.891	**0.009**	9.198	**0.009**	7.224	0.011	**6.113**	0.026
		shadow	6.766	**0.008**	9.029	0.013	6.772	0.009	**6.498**	0.115
	70°	normal	6.462	0.012	8.798	**0.011**	6.854	**0.011**	**5.701**	0.095
		shadow	6.252	**0.008**	8.525	0.010	6.166	0.012	**5.722**	0.126
	80°	normal	5.230	0.022	7.722	**0.016**	5.917	0.021	**4.675**	0.163
		shadow	5.400	0.014	7.342	**0.011**	5.399	0.019	**4.824**	0.143
piCam	0°	normal	1.451	**0.024**	2.599	0.025	1.972	0.048	**0.521**	0.105
		shadow	1.592	**0.034**	2.603	0.056	1.617	0.060	**0.490**	0.105
	10°	normal	2.541	0.033	3.743	**0.013**	3.268	0.031	**2.280**	0.234
		shadow	2.636	0.022	3.630	0.048	**2.340**	**0.014**	2.446	0.102
	20°	normal	3.636	0.029	4.664	0.015	4.030	0.041	**2.639**	**0.014**
		shadow	3.963	0.019	4.636	0.032	3.883	0.019	**2.608**	**0.015**
	30°	normal	4.832	0.030	5.474	**0.016**	4.972	0.032	**3.580**	0.023
		shadow	4.996	0.019	5.557	**0.018**	4.808	0.021	**3.499**	0.020
	40°	normal	5.419	0.024	6.284	**0.012**	5.886	0.026	**4.393**	0.024
		shadow	5.652	0.018	6.528	0.025	5.739	**0.011**	**4.426**	0.021
	50°	normal	6.137	0.030	6.716	**0.013**	6.491	0.031	**4.927**	0.030
		shadow	6.181	0.024	7.004	**0.017**	6.236	0.026	**4.923**	0.029
	60°	normal	6.322	0.039	6.991	**0.016**	6.765	0.032	**5.148**	0.067
		shadow	6.389	0.028	7.216	**0.016**	6.460	0.023	**5.249**	0.030
	70°	normal	6.021	0.050	7.031	**0.016**	6.895	0.027	**5.198**	0.143
		shadow	6.412	0.030	7.252	0.030	6.421	**0.027**	**5.306**	0.035
	80°	normal	5.801	**0.006**	6.637	0.015	6.431	0.029	**4.984**	0.060
		shadow	**5.171**	0.213	6.998	0.031	5.954	**0.020**	5.336	0.430

Angle Data - Distance Results: Presents the calculated distance of the marker when the camera is kept at a distance of $1m$ from the target and angle is varied from $0°$ to $80°$. Angle is measured in degrees, mean and standard deviation (std) are measured in cm, and rate is measured in percentages. Detection rate is omitted here since it is the same as in Table III. The package with the lowest error for each angle and lighting condition is denoted in blue and the lowest standard deviation is noted in magenta.

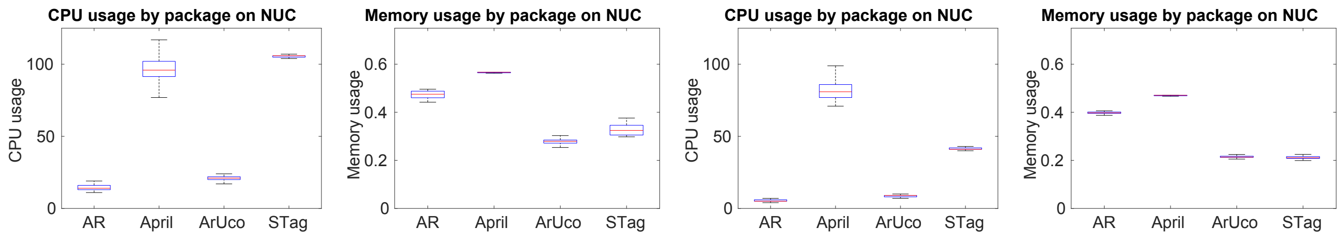
When processing higher resolution images, both the CPU and memory usage increase on the NUC and the TX2. The most significant change is the CPU usage of STag on the NUC where the high resolution images require more than double the processing power of the low resolution images.

V. Conclusions

In this work, we present experimental data from four different fiducial marker packages. The experiments show how the relative position and orientation of the marker with respect to the camera and lighting conditions affect the accuracy and stability of the pose measurements as well as the detection rate. Moreover, the computational cost for each package on three different computers commonly used in robotic and UAS applications is presented.

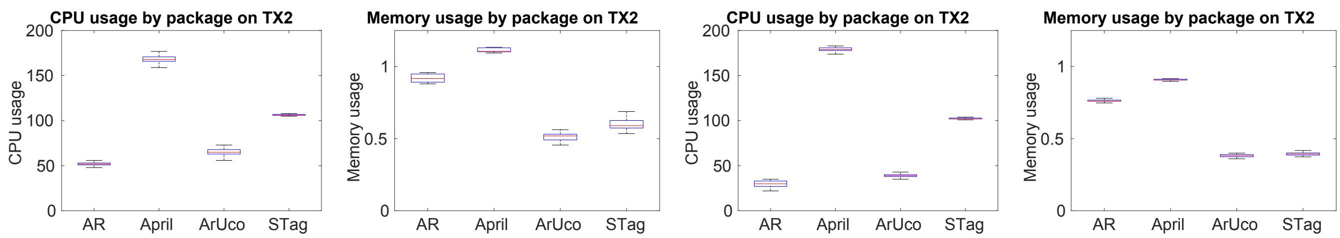
Based on the results of this work, AprilTag, ArUco and STag all present high detection rates in almost all settings tested. STag has the best performance when it comes to position measurements and AprilTag shows the best results in orientation measurements. In both cases however, ArUco is consistently the second best package. While STag was expected to present the most stable results, AprilTag and ARTag present comparable stability with AprilTag presenting higher stability in orientation measurements. The fact that the ROS implementation developed uses a different algorithm, than the one proposed by the original paper, to find the camera pose from the detected corners of the marker, might have affected the overall stability of the results. In the future, we aim to augment the ROS implementation to add the proposed algorithm for single marker detection and use solvePnP for marker bundles.

It is also worth mentioning that in all cases, the low resolution camera with the higher contrast produced images



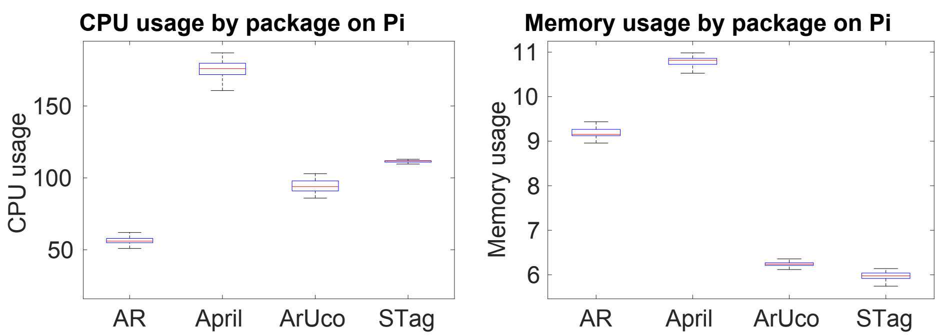
(a) (b) (c) (d)

Fig. 5: Processing cost on the NUC platform. Figures (a) and (b) show the CPU and memory usage for the high resolution images, respectively, while (c) and (d) show the CPU and memory usage for the low resolution images, respectively. The CPU usage is reported as the percentage of a single CPU core of the platform.



(a) (b) (c) (d)

Fig. 6: Processing cost on the NVidia TX2 platform. Figures (a) and (b) show the CPU and memory usage for the high resolution images, respectively, while (c) and (d) show the CPU and memory usage for the low resolution images, respectively. The CPU usage is reported as the percentage of a single CPU core of the platform.



(a) (b)

Fig. 7: Processing cost on Raspberry Pi. Figures (a) and (b) show the CPU and memory usage for the low resolution images, respectively. The CPU usage is reported as the percentage of a single CPU core of the platform. The high resolution data are omitted because detection rates failed to exceed 1Hz, making their implementation impractical.

which yielded the lowest errors. The only advantage of the high resolution images was the increased detection rate in all cases except ARTag.

In terms of computational efficiency, AprilTag is the package that requires the most resources both on CPU and memory usage. STag has the second highest CPU usage but has one of the lowest memory usages. ARTag and ArUco are the most computationally efficient packages, although ArUco presents a superior performance on measurement accuracy and detection rate. On low-power and low-cost platforms such as the Raspberry Pi, ArUco is the most suitable choice. On more powerful computers though, all packages except ARTag present viable choices. Depending on the application requirements, any of the three packages can be the preferred solution.

References

[1] Q. Feng, J. Liu, and J. Gong, “UAV Remote Sensing for Urban Vegetation Mapping Using Random Forest and Texture Analysis,” *Remote Sensing*, vol. 7, no. 1, pp. 1074–1094, Jan. 2015.

[2] T. Duan, P. C. Hu, and L. Z. Sang, “Research on route planning of aerial photography of UAV in highway greening monitoring,” *Journal of Physics: Conference Series*, vol. 1187, no. 5, p. 052082, Apr. 2019.

[3] D. Zhang, K. Burnham, L. McDonald, C. MacLeod, G. Dobie, R. Summan, and G. Pierce, “Remote inspection of wind turbine blades using UAV with photogrammetry payload,” *56th Annual Conference of*

the British Institute of Non-Destructive Testing, NDT 2017, pp. 1–11, 2017.

[4] G. Chahine and C. Pradalier, “Survey of Monocular SLAM Algorithms in Natural Environments,” in *2018 15th Conference on Computer and Robot Vision (CRV)*. IEEE, May 2018, pp. 345–352.

[5] B. Joshi, S. Rahman, M. Kalaitzakis, B. Cain, J. Johnson, M. Xanthidis, N. Karapetyan, A. Hernandez, A. Q. Li, N. Vitzilaios, and I. Rekleitis, “Experimental Comparison of Open Source Visual-Inertial-Based State Estimation Algorithms in the Underwater Domain,” in *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, Nov. 2019, pp. 7227–7233.

[6] S. Kyristsis, A. Antonopoulos, T. Chanialakis, E. Stefanakis, C. Linardos, A. Tripolitsiotis, and P. Partsinevelos, “Towards Autonomous Modular UAV Missions: The Detection, Geo-Location and Landing Paradigm,” *Sensors*, vol. 16, no. 11, p. 1844, Nov. 2016.

[7] M. Zurn, K. Morton, A. Heckmann, A. McFadyen, S. Notter, and F. Gonzalez, “MPC controlled multirotor with suspended slung Load: System architecture and visual load detection,” in *2016 IEEE Aerospace Conference*. IEEE, Mar. 2016, pp. 1–11.

[8] D. Pickem, L. Wang, P. Glotfelter, Y. Diaz-Mercado, M. Mote, A. Ames, E. Feron, and M. Egerstedt, “Safe, Remote-Access Swarm Robotics Research on the Robotarium,” *arXiv:1604.00640*, Apr. 2016.

[9] A. W. Yew, S. K. Ong, and A. Y. Nee, “Immersive Augmented Reality Environment for the Teleoperation of Maintenance Robots,” *Procedia CIRP*, vol. 61, pp. 305–310, 2017.

[10] B. Benligiray, C. Topal, and C. Akinlar, “STag: A stable fiducial marker system,” *Image and Vision Computing*, vol. 89, pp. 158–169, Sept. 2019.

[11] M. Quigley, B. P. Gerkey, K. Conley, J. Faust, T. Foote, J. Leibs, E. Berger, R. Wheeler, and A. Y. Ng, “ROS: an open-source Robot Operating System,” Tech. Rep., 2009.

[12] D. B. dos Santos Cesar, C. Gaudig, M. Fritsche, M. A. dos Reis, and F. Kirchner, “An evaluation of artificial fiducial markers in underwater environments,” in *OCEANS 2015 - Genova*. IEEE, May 2015, pp. 1–6.

[13] M. Fiala, “ARTag, a Fiducial Marker System Using Digital Techniques,” in *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR’05)*, vol. 2. IEEE, 2005, pp. 590–596.

[14] ——, “Designing Highly Reliable Fiducial Markers,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 32, no. 7, pp. 1317–1324, July 2010.

[15] E. Olson, “AprilTag: A robust and flexible visual fiducial system,” in *2011 IEEE International Conference on Robotics and Automation*. IEEE, May 2011, pp. 3400–3407.

[16] J. Wang and E. Olson, “AprilTag 2: Efficient and robust fiducial detection,” in *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, Oct. 2016, pp. 4193–4198.

[17] S. Garrido-Jurado, R. Muñoz-Salinas, F. Madrid-Cuevas, and M. Marín-Jiménez, “Automatic generation and detection of highly reliable fiducial markers under occlusion,” *Pattern Recognition*, vol. 47, no. 6, pp. 2280–2292, June 2014.

[18] J. DeGol, T. Bretl, and D. Hoiem, “ChromaTag: A Colored Marker and Fast Detection Algorithm,” in *2017 IEEE International Conference on Computer Vision (ICCV)*. IEEE, oct, pp. 1481–1490.

[19] H. Kato and M. Billinghurst, “Marker tracking and HMD calibration for a video-based augmented reality conferencing system,” in *Proceedings 2nd IEEE and ACM International Workshop on Augmented Reality (IWAR’99)*. IEEE Comput. Soc, 1999, pp. 85–94.

[20] J. Bacik, F. Durovsky, P. Fedor, and D. Perdukova, “Autonomous flying with quadrocopter using fuzzy control and ArUco markers,” *Intelligent Service Robotics*, vol. 10, no. 3, pp. 185–194, July 2017.

[21] A. Babinec, L. Jurišica, P. Hubinský, and F. Duchoň, “Visual Localization of Mobile Robot Using Artificial Markers,” *Procedia Engineering*, vol. 96, pp. 1–9, 2014.

[22] H. Lim and Y. S. Lee, “Real-Time Single Camera SLAM Using Fiducial Markers.” Fukuoka, Japan: IEEE, 2009, p. 177.

[23] J. Chen, T. Liu, and S. Shen, “Tracking a moving target in cluttered environments using a quadrotor,” in *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, Oct. 2016, pp. 446–453.

[24] S. Chaves, R. Wolcott, and R. Eustice, “NEEC Research: Toward GPS-denied Landing of Unmanned Aerial Vehicles on Ships at Sea,” *Naval Engineers Journal*, vol. 127, no. 1, pp. 23–35, 2015.

[25] O. Araar, N. Aouf, and I. Vitanov, “Vision Based Autonomous Landing of Multirotor UAV on Moving Platform,” *Journal of Intelligent & Robotic Systems*, vol. 85, no. 2, pp. 369–384, Feb. 2017.

[26] P. Vlantis, P. Marantos, C. P. Bechlioulis, and K. J. Kyriakopoulos, “Quadrotor landing on an inclined platform of a moving ground vehicle,” in *2015 IEEE International Conference on Robotics and Automation (ICRA)*. Seattle, WA, USA: IEEE, May 2015, pp. 2202–2207.

[27] M. Kalaitzakis, B. Cain, N. Vitzilaios, I. Rekleitis, and J. Moulton, “A marsupial robotic system for surveying and inspection of freshwater ecosystems,” *Journal of Field Robotics (Accepted)*, 2020.

[28] A. Borowczyk, D.-T. Nguyen, A. Phu-Van Nguyen, D. Q. Nguyen, D. Saussié, and J. L. Ny, “Autonomous Landing of a Multirotor Micro Air Vehicle on a High Velocity Ground Vehicle,” *IFAC-PapersOnLine*, vol. 50, no. 1, pp. 10 488–10 494, July 2017.

[29] G. Schweighofer and A. Pinz, “Robust Pose Estimation from a Planar Target,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 28, no. 12, pp. 2024–2030, Dec. 2006.