Title

Pengju Jin, Siddhartha Srinivasa

***Abstract*—Abstract Goes here**

I. Introduction

Detection and identification using artificial landmarks, known as fiducial markers, has long been used in augmented reality and computer vision applications. Over the last decade, there has been numerous marker systems, such as ARTags, Apriltags, and Rune Tags, designed to improve detection encoding precision. However, these systems are not robust enough to be used reliability in many robotic applications.

Compared to markerless detection algorithms, these fiducial marker methods are simpler and more consistent. There has been significant effort in improving the detection speed and encoding accuracy. They have yield great results for computer vision tasks that require high detection accuracy like camera calibrations, 3D reconstruction. Furthermore, they have gained popularity in the robotic community for having unique characteristics of high detection rates and numerous encoding schemes. For example, Apriltags are commonly used to test SLAM systems or finding ground truth for objects in manipulation and motion planning tasks as shown in Figure ??.

Despite all the improvements, obtaining accurate pose estimations from these tags remain a challenge. This is especially important for robotic applications because small errors can cause large system failures as the errors propagate and amplify through the system. Currently, these systems yield good results under well conditioned or rendered environments, but this does not translate to ill-conditioned settings. For instances, when Apriltags are used with low resolution camera or harsh lighting conditions, the system often produce poses with tremendous rotational errors as shown in Figure 3. In fact, we observe that the localization accuracy of this system perform significantly worse at difficult viewing angles or when there are noise in the scene. This is a difficult problem because RGB sensors are sensitive to lighting and current fiducial systems are not designed to take advantage of other sensors commonly available on robots.

We present two main contributions in this paper. First, we conducted an in-depth analysis on the effect of various noises on the pose estimation process. In particular, the noise in RGB images creates a perspective ambiguity problem that makes the pose estimation challenging without additional information. Second, we describe a novel method that takes advantage of the RGBD sensors that are commonly available on most robotic systems to accurately estimate the pose from a single tag under noisy conditions in real time. In the core of the algorithm, we recognize that RGB and depth sensors work optimally under different conditions. Therefore, their strength can be combined meaningfully to improve the localization accuracy robust to illumination. There are few key features to this algorithm:

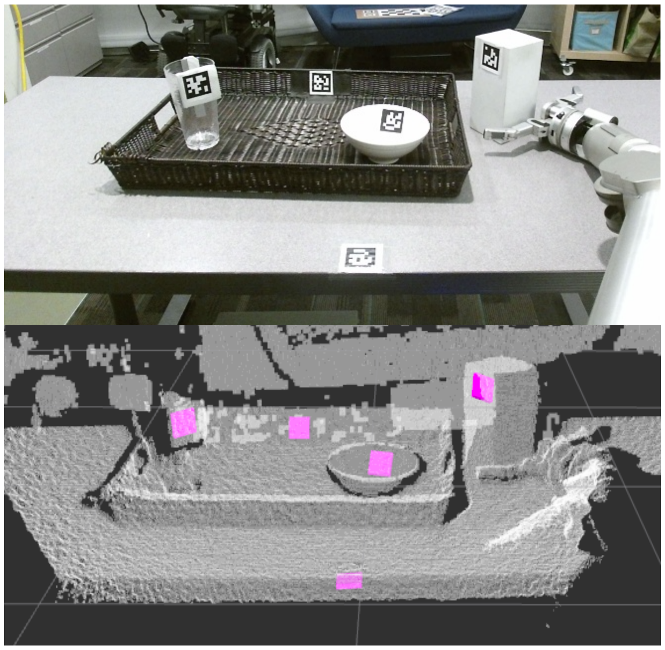


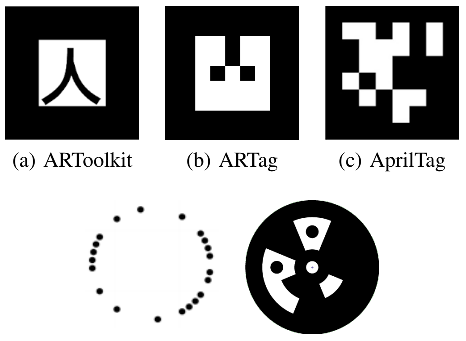
Fig. 1: Apriltag used to localize objects in mainpulation tasks

- This method is highly robust to noise in the scene. It can obtain accurate poses suitable for a wide range robotic applications.
- It is easily generalizable to most fiducial tags designs.
- It performs at worse at least as good as using only RGB images.
- It has very small computation overhead and can be ran in real time.

This paper also presents empirical results demonstrating the the successful performance of the algorithm on captured data from an humanoid robot. Our implementation of the algorithm is based off of the Apriltag detection pipeline and it is integrated with ROS.

II. Related Work

Be able to obtain highly accurate pose estimation has been an important research area in robotics. There are numerous algorithms developed that rely only on RGB or grey scale images. Most of these methods rely on solving the projection geometry of some detected features and then minimize the reprojection error of the features in the image space []. Similarly, methods such as Iterative Closet Point were developed to solve the pose estimation problem using range data by minimizing the Euclidean distance between



(a) ARToolkit (b) ARTag (c) AprilTag

(d) RUNE-Tag (e) Intersense

Fig. 2: Different types of popular fiducial tags. ARToolkit, ARTags, and AprilTags are square tags with black borders. RUNE-tags and Intersense use different circle features as landmarks

the model and the depth data. Recently, there are some approaches proposed to enhance the accuracy of traditional algorithms by fusing RGB and depth data in various problems by using extended Kalman filter []. Compared to the single sensor approaches, algorithms utilizing RGBD data are more accurate and performs well in noisy situations where other approaches fail.

Using fiducial marker is an example of pose estimation methods exploiting easily detectable features in the RGB space. Although there is an abundance of unique tag designs. most of them carry easily recognizable yet precise binary patterns in the inner region to encode information with. There are two large type of common tags: square tags and circle tags.

Circular tags are created to encode the payload using small circle patterns arranged in various shapes. The example of circular tags include Intersense[] and Rune tags[]. The perspective transformation of a circle is an ellipse, which can be used to directly compute the pose using back projection methods. The detection of circular features are generally more accurate to localize and thus generates better pose estimations at the cost of higher computation time. However, small circular features become hard to detect when they are far away from the camera or prospectively rotated and thus their effective range is much smaller than that of square tags. This characteristic making them less useful in applications with size constraints.

ARTags[], ARToolkit[], AprilTag[] and AprilTag 2[] are examples of squared based fiducial tags. The perspective projection of a square becomes a general quadrilateral which can be computed easily using any contour tracing algorithm. Given the scale of a single marker, the full 6-DOF pose can then be estimated using the corners of the quadrilateral. However, since they are detected using rectangles and lines, the accuracy of their corner point sub-pixel locations are limited. Among the square tags, ARToolkit is one of the earliest detection system and it was mainly used for Augmented reality applications. Instead of using a binary payload, it used various symbols to encode the tag and it was computationally expensive to decode the tag. Built on top of ARToolkit, ARTags and Apriltag reduced the computation time by using a 2D binary pattern as the payload. Both systems use the image gradient to compute the tag border making it robust to lighting changes and partial occlusions. Relative to the ARTags, Apriltag has a lower false positive rate as it uses a lexicode-based system that is invariant to rotation. In addition, the Apriltags have higher detection rates at further distances and at more difficult viewing angles. Recently AprilTag 2, improved upon the origin Apriltag, implemented a new boundary segmentation algorithm which further reduced the computing time for detection and increased the detection rate. Compared to circular tags, the advantages of the square tags are that they can be located very efficiently and they have reliable decoding schemes. Therefore, even though the square tags have slightly lower localization accuracy, they are more suitable for robotic applications that require a robust system.

III. Challenges

In square fiducial marker detection, the pose is calculated by using the four corners of the tag. Since the tags are planar, it is easy to compute perspective point correspondences from the corners. This can be formalized as a specific case of the Perspective-N-Point problem and it has been well studied in geometry based Computer Vision literatures [][]. There are numerous optimization methods such as ones purposed in [] and [] to solve this problem. In particular, in [], the author shows that there is a deterministic solution to the Perspective-4-Point (P4P) problem. In other words, given the projection of a tag's 4 corners, the pose of the tag is unique. In practice, however, when a tag is captured in a low resolution camera, this method is very sensitive to noise. For instances, when ARTags, Apriltags and ARToolkit systems are used in scenarios shown in Figure **??**, the poses of the tags alter even when the scene is static. Since the minimal number of perspective points are used to estimate the pose, a small variance in the corner detection process will yield estimations far from the true pose as shown in Figure 3.

We will illustrate an ambiguity effect caused by noise by using two over lapping cubes in Figure 4. The overlapping face of the two cubes are interlaced but rotated by 120 degrees. However, due to perspective projection, the squares appears to be on the same plane. Under low camera resolution, the over lapping squares become virtually indistinguishable. The red circular regions are the detected corners under some sensory noise. Even though the reprojection error is minimized in the 2D space using P4P optimization methods, the 3D pose can still be far off. The result of the optimization can be characterized as a bimodal distribution and a function of the the viewing angle. Depending on the noise level in the scene, the optimization might converge to either one of the local minimum causing the pose estimation to be unreliable.

IV. Approach

This section describes a method for accurately estimating poses for square fiducial tags in noisy settings by fusing RGBD data. The process of detecting and decoding the tag

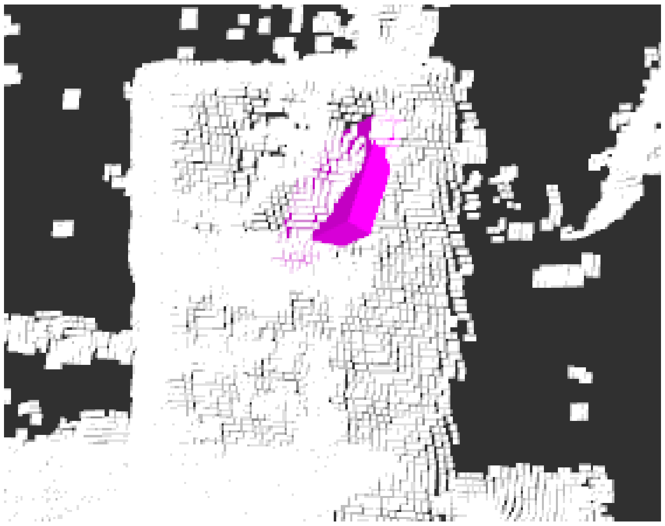


Fig. 3: The orientation of Apriltag placed on the object is greatly misaligned with the actual object, making the robot nearly impossible to grab it

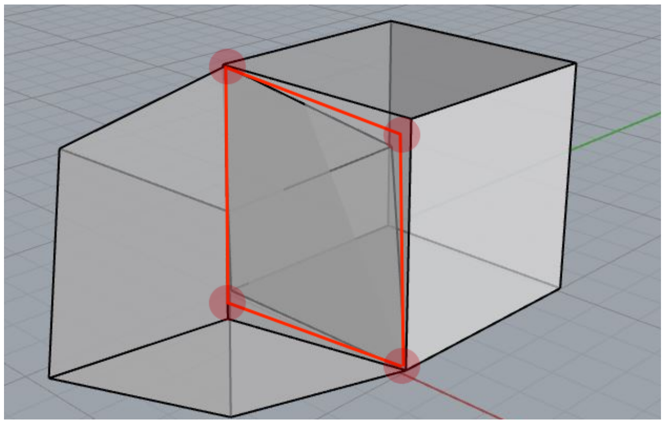


Fig. 4: Perspective Ambiguity illustrated with two overlapping cubes. The red square is a simulated projection of a tag and the orientation of the square is

is identical to previous methods. This also make our method easily generalizable to any square tag detection systems. However, the detected corners are treated as approximated locations of the true corners. Using the corners, the method implicitly evaluate the depth data and RGB data as two independent observations and fuse them to minimize the error in 2D and 3D space.

There are three distinct components to this method. First, we find the plane in $SO(3)$ containing the tag using depth data and detected corners. Secondly, an approximate initial pose is computed using the depth plane. Finally, the method refines the initial pose using the RGB data by minimizing the reprojection error within a constrained space. Each component is described in detail in the following subsections.

A. *Depth Plane Fitting*

With a calibrated RGBD sensor, depth and RGB streams are registered to the same frame. The square patch of points in the depth image defined by the g also contains the range information of the tag. Here we take advantage of the planar characteristic of the tag. By fitting a plane over the range data, we can constrain the pose of the tag to be on the plane.

The raw range data retrieved from the Kinect One sensor are usually noisy as illustrated by Figure 5. Specifically, borders of the tag and some of the dark regions of the tag produce highly unreliable range data. Therefore, we first filter the data by removing points too far from the median before fitting the plane. Nevertheless, the remaining points could have a large variance depending on the lighting condition and the magnitude of the plane rotation. The accuracy of the plane and tag pose is directly affected by the noise level of data. In fact, we want to characterize the uncertainty and weight the initial estimation accordingly during the fusing stage.

In implementation, we used a Bayesian plane fitting algorithm described in [Uncertainty Analysis] which computes the Hessian Normal parameters of a plane through optimizing

$$\min_{\hat{\boldsymbol{n}},d} \sum_{j=1}^{N} \frac{(p_j(\hat{\boldsymbol{n}} \cdot \hat{\boldsymbol{m}}_{\boldsymbol{j}}) - d)^2}{(\hat{\boldsymbol{n}} \cdot \hat{\boldsymbol{m}}_{\boldsymbol{j}})^2 \sigma^2\{\bar{p}_j\}} \tag{1}$$

The algorithm in the paper assumes a radial Gaussian noise in the range data p_j with the standard deviation modeled by a function in the form

$$\sigma\{\bar{p_j}\} = \frac{kd^2}{\|\hat{\boldsymbol{n}} \cdot \hat{\boldsymbol{m}}_{\boldsymbol{j}}\|} \tag{2}$$

where $\hat{\boldsymbol{n}}$ is the local normal to the planar surface of the depth point and $\hat{\boldsymbol{m}}_{\boldsymbol{j}}$ is the measurement direction for the sensor for point p_j. The coefficient $k > 0$ is an estimated value obtained from sensor calibration. In our implementation, we obtained k by simplified the results from [Kinect Noise Model paper].

Another important result we used from [] is the covariance matrix for the plane-parameters. The covariance matrix is obtained by taking the *Moore-Penrose generalized inverse* of Hessian matrix computed from the Lagrangian. The covariance characterizes the uncertainty of the plane and defines the refinement regions of the pose later in the pipeline.

B. *Initial Pose Estimation*

The pose of the tag can be described as the transformation from tag frame's coordinate system to the sensory frame's coordinate system of the robot given by $[R, \boldsymbol{t}]$. However, the depth plane $D[\hat{\boldsymbol{n}}, d]$ alone is insufficient to determine the transformation because it only defines 3 DOF in a 6 DOF space. Furthermore, the center of the tag must align with the center of the plane and thus further constrain 2 DOF. However, there are still infinite number of valid poses rotating about the normal $\hat{\boldsymbol{n}}$. One solution is to constrain the pose by using a corner as an extra point correspondence to solve for the optimal rotation. However, the accuracy of this method largely depends on the depth accuracy of the chosen corner point.

An alternative is to use all 4 detected corners as 4 pairs of point correspondences for the optimization. We projected the detected corners onto $D[\hat{\boldsymbol{n}}, d]$ to get the coordinates $\boldsymbol{p} = [p_1, p_2, p_3, p_4]$ in the robot sensory frame. The corner coordinates $\boldsymbol{q} = [q_1, q_2, q_3, q_4]$ in the tag frame can be easily calculated since the tag is a square plane. We define the center of the tag as the origin, the corner coordinates are simply the

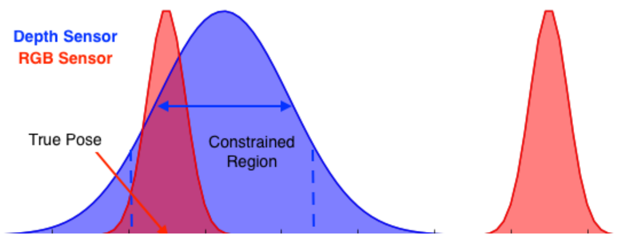


Fig. 5: An abstract visualization of the optimization constraints. The blue curve is the initial pose estimation obtained from the depth plane. The red curves are the ambiguous poses from the RGB image. We constrained the region of optimization based on how well we fit the depth plane.

location of the corners on a Cartesian plane. Given two sets of 3D point correspondences, this can be computed as a rigid body transformation estimation [Reference to SVD 3D rigid body]. Solving for the optimal transformation $[R, \boldsymbol{t}]$ requires minimizing a least squares error objective function given by:

$$[R, \boldsymbol{t}] = \underset{R \in SO(3), \boldsymbol{t} \in \mathbb{R}^3}{\operatorname{argmin}} \sum_{i=1}^{n} w_i \| R\boldsymbol{q_i} + \boldsymbol{t} - \boldsymbol{p_i} \|^2 \tag{3}$$

There are numerous approaches to solve Eq. 3. Since we have very few correspondences and they are assumed to be correct, it can be computed quickly using SVD:

$$\bar{p} = \frac{1}{N} \sum_{i=1}^{N} p_i \qquad p_{ci} = p_i - \bar{p} \tag{4}$$

$$\bar{q} = \frac{1}{N} \sum_{i=1}^{N} q_i \qquad q_{ci} = q_i - \bar{q} \tag{5}$$

$$p_c^\top q_c = U \Sigma V^\top \tag{6}$$

$$R = V U^\top \tag{7}$$

$$\boldsymbol{t} = \bar{q} - R\bar{p} \tag{8}$$

R and t are the corresponding rotation and translation components of the the transformation. The above approach minimizes the least square error of the transformation and it is robust to small errors in the correspondences. The resulting pose obtained from the range data, although not accurate, is robust to noise and provide a good approximation for the true pose.

C. Pose Refinement

Lastly, the pose is refined by minimizing the reprojection error using the initial pose estimated from the previous step. The camera is assumed to be calibrated and the camera projection model K is known. We denote R^* and $\boldsymbol{t}^*$ to be the optimal pose in the constrained optimization function

$$[R^*, \boldsymbol{t}^*] = \underset{R^*, \boldsymbol{t}^*}{\operatorname{argmin}} \sum_{i}^{n} \| (K[R^* | \boldsymbol{t}^*]) \boldsymbol{p_i} - \boldsymbol{y_i} \|^2 \tag{9}$$

$$R^* = R(\Delta R) \tag{10}$$

$$\boldsymbol{t}^* = \boldsymbol{t} + R(\Delta \boldsymbol{t}) \tag{11}$$

subject to:

$$\Delta R < \Gamma_R \tag{12}$$

$$\Delta \boldsymbol{t} < \Gamma_t \tag{13}$$

Intuitively, the most optimal pose is the one with minimal reprojection error in the RGB space and align with the plane in the depth space. Therefore, the goal of the optimization is to find the local minimum closest to the initial estimation within allowable region Γ as illustrated with Figure 6. The key challenge is to determine the constrained region Γ_R and Γ_t such that it include a locally optimal pose and exclude the ambiguous pose. In most cases where the depth plane yields a good fit, this region should be small because the optimal pose is close to the initial estimate. When the depth sensor is noisy, the Γ increases since the initial estimate might be far off. Thus, the constrained region Γ is defined by the uncertainty in the initial estimate and it is characterized by the covariance of the plane parameters. In implementation, we used a trust-region optimization algorithm to bound the constraints. The scaling parameters for the covariance is empirically test to obtain the best results for our robot.

The strength of this method is that it harness the benefits of RGB and depth information without explicitly assuming their relative accuracy. One advantage of RGBD sensors is that the camera and the depth sensor often work optimally with different constraints. In the example of Kinect, the RGB camera is sensitive to lighting and works poorly in scenes with low illumination. However, the time of flight depth sensor are unaffected by such a problem. On the hand, the time of flight sensor yield poor range results on surface edges, but the RGB camera works exceptionally well with edges where there is a high color contrast. Therefore, our proposed method can work well in a wide range of conditions making it robust for robotic applications.

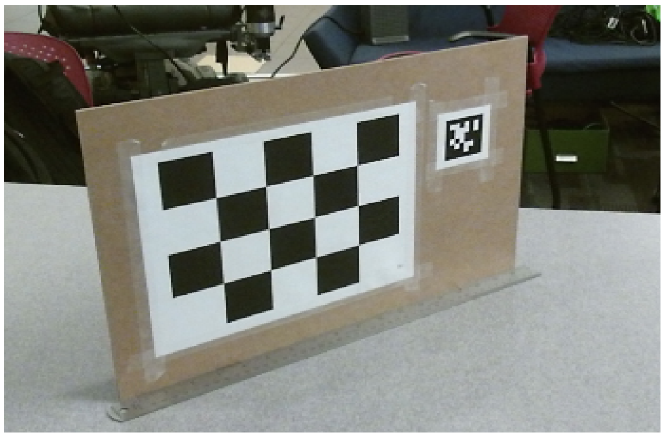
V. Experimental Results

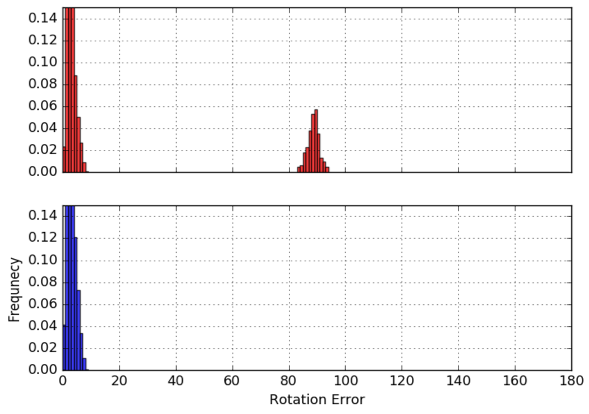
The key problem we are trying to resolve is the localization accuracy of Apriltags in noisy situations. There are two major components we want to demonstrate in this paper: first, we want to characterize the effect of perceptual ambiguity and noise on the Apriltag detection algorithms. Second, we want to test the resilience of our algorithm and show that it can obtain reasonable pose estimations under high level of noise. Finally, we briefly tested the runtime of the algorithm to show that it remains capable of real time detection.

In our experiments, we measured the rotational and translational accuracy of the detections algorithms with respect to three different independent variables: viewing angles, distances, and lighting conditions. We placed a standard camera calibration chessboard and an Apriltag of known size on a solid planar board. The Apriltag has a fixed distance from the chessboard. This is used to compute the ground-truth pose for the tag. By using a large chessboard, we can detect the corners to a sub-pixel accuracy and compute accurate ground-truth poses unsusceptible to lighting and sensory noises. Figure 7 demonstrates the experimental set up.

A. Viewing Angle

The low localization accuracy caused by the perceptual ambiguity of the Apriltags is a non-linear function on the


(a) RGB Image at 40°



(b) Close

Fig. 6: The fiducial tag plane captured by the depth sensor

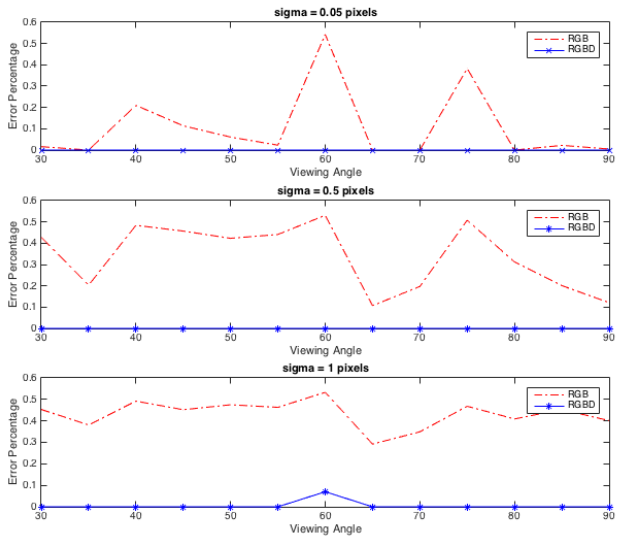


Fig. 7: Viewing Angle vs Error Percentage under different simulated noise level. The new RGBD based algorithm can resist noise in the RGB image and it vastly outperforms the original algorithm.

viewing angle of the tag. To characterize the effect, we placed the testing board on a table straight in front of the robot in a well lit room. Since the sensor is taller than the plane of the table, the robot has to slightly gazing down at it. We rotated the testing board at a increment of 5 degrees from 30 degrees to 90 degrees. The angles are measured from the axis perpendicular to gazing direction of sensor. This is about the range in which the tag can be detected reliably given the camera resolution and distance. At each angle, we captured the RGB image, depth image, and detection outputs from the Apriltag algorithm.

For each captured data collection, we introduced 3 levels of Gaussian noise of $\sigma = 0.2$, $\sigma = 0.5$, $\sigma = 1$ to the RGB image and computed the resulting tag pose. This is repeated for 1000 trails at each noise level and the errors are computed for each trial. Figure 8 shows some of the results.

As the result shown in figure 7 indicates, the viewing angle

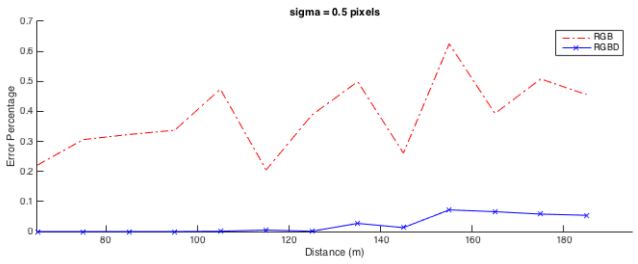


Fig. 8: Distance vs Error Percentage. Data are captured at a 10 cm increment from 65 cm to 185 cm.

has large effect on the rotation error. As we expected, the empirical results show a very clear bimodal distribution for the Apriltags at different viewing angles. The depth-sensor fused algorithm vastly outperforms the previous algorithm as it is not affected by the perceptual ambiguities. The small amount of the noise introduced to the data only cause a small rotational change around the true pose of the tag. In Figure[8], we threshold all the poses based on their rotational errors and plotted the percentage of unacceptable poses at each viewing point. One interesting observation from the data is that, at most viewing angles, the magnitude of noise above a certain threshold has little effect on the localization accuracy. At most viewing angles, relatively small noises causes a significant accuracy decrease.

B. Distance

To test the estimation accuracy with respect to distance, we captured the images at different distances away from the camera at a fixed angle. Figure 9 shows the experiment results.

The relationship between the distance and localization accuracy is much more apparent. As the tag moves further away from the sensor, the number of pixels on the tag decreases. The perspective ambiguity effect becomes more apparent when there is only a small patch of pixels on the tag. However during the experiment, it is difficult to keep the viewing angle precisely consistent at each trail. Therefore, the

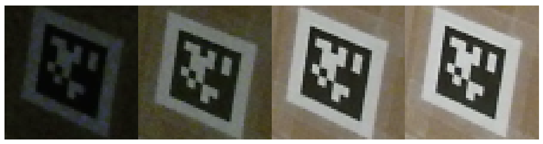

(a) Dark (b) Dim (c) Normal (d) Bright

Fig. 9: Apriltags captured by Kinect V2 under different levels of illumination. The RGB sensor dynamically adjust the exposure time to compensate for low lighting. In (a), the image is captured outside of Kinect's adjustable range and the pixels are underexposed. In (b), the long exposure time introduced noticeable noise to the image.

pose error percentage using RGB is not increasing smoothly in Figure 9.

We see a clear increase in error percentage in the proposed method when the tag is far away from the camera. This is contributed both by a smaller tag patch size in the depth image and an increasing in noise with the Kinect sensor at a further distance. In these cases, the variance of the depth plane estimation became very wide and the algorithm is unable to converge to the correct pose. The Nevertheless, our method shows a significant gain in accuracy at every distance.

C. Lighting

From our past observations, poor lighting condition is the most significant contributing factor to noise and it results in low localization accuracy. The Kinect V2 sensor used in our experiments dynamically adjust the exposure time under low lighting conditions. When pictures are taken below or near the adjustable range of the sensor, they contain very noticeable noise as shown in Figure **??**.

We also tested the algorithm under harsh lighting conditions in a real world setting. The data were captured under 4 different lighting conditions: 20 lux (dark), 43 (lux) dim, and (90 lux) normal, 243 lux (bright). We recorded a static scene over 5 seconds and randomly sampled 100 frames to run the test. As results shown in Figure 11, the localization accuracy significantly improves with better illumination. At the lowest illumination, nearly 25% of the poses were unacceptable (more than 20 degrees off) at the particular viewing angle. By using depth sensor which is unaffected by poor source radiance, there are only 3% of unacceptable poses.

D. Benchmark Against ar_track_alvar

ar_track_alvar is a ROS wrapper package for Alvar [], an open source AR tag tracking library. The package is designed for AR tag detection and pose estimation for robots similar to Apriltags. In particular, it implements a module where depth sensor is integrated to improve the pose estimation. The package use the detected corner points to extract a patch of point clouds containing the tag. Then it proceed to fit a plane of the point cloud data and find its centroid using PCL. The pose of the tag is computed by aligning the centroid with the center of the tag.

We implemented a similar module for the Apriltag and compared the pose accuracy between our proposed method and the module using all the collected data. The results are shown in Figure 12. The two algorithms performed similarly

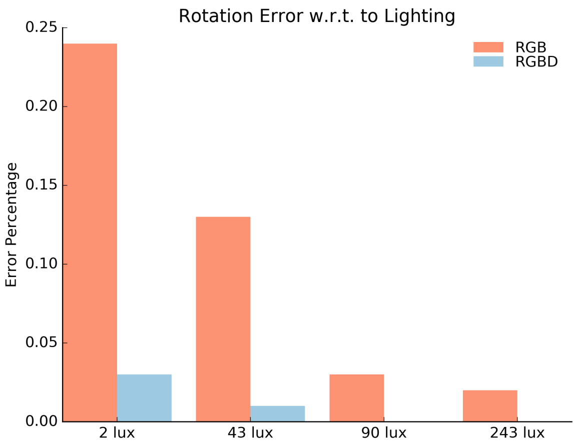
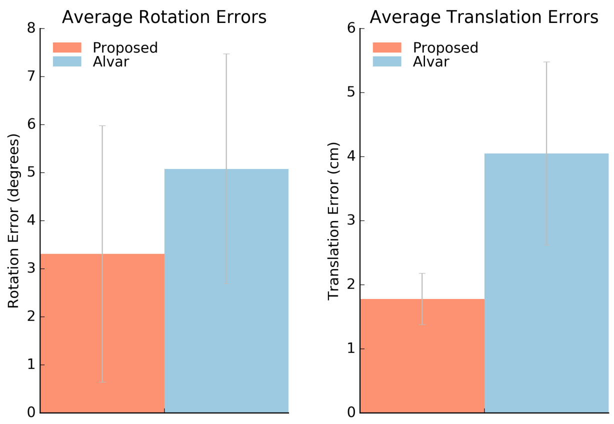


Fig. 10: Illumination vs Error Percentage. Data are captured at 65 cm away from the camera at a 40 degree angle.



(a) Rotation Error (b) Translation Error

Fig. 11: Average pose errors compared with ar_track_alvar package.

in rotation error, but the proposed method was on average 2 cm better with the position component. The spread of error is also much smaller for the position component indicating that our purposed method is more consistent.

E. Computation Time

We briefly tested the computation time of the new algorithm. With our current implementation in Python, the additional computation time for sensor fusing process is 11 ms. Therefore the entire detection pipeline can process a 960 x 540 image within 35 ms. All tag detectors and the fusing process were running in a single-threaded mode of an Intel core. Since our sensory updates at roughly $35Hz$, the entire pipeline can process the tags and estimate the pose in near real time. There is no significant time increase on a higher resolution image for the fusing process because our algorithm does not need to process the entire image.

The most time consuming step is running the trust region optimization for refining the pose. This process can be sped up significantly by simply implementing the pipeline in C++.

VI. Conclusion

In this paper, we did a in depth analysis of the localization problem with Apriltags. We proposed a novel algorithm of using RGBD sensors to accurately compute the pose of Apriltags robust to noise. It is particularly suitable for robotic applications which requires precise poses such as manipulation, SLAM, and others. Furthermore, this technique can be easily generalized to other types of planar fiducial tags. Our implementation is fully open sourced and available at:

http://somegithublink.com