18th International Conference on Information Fusion
Washington, DC - July 6-9, 2015

RGBD Data Based Pose Estimation: Why Sensor Fusion?

O. Serdar Gedik
Department of Computer Engineering,
Yildirim Beyazit University,
Ankara, Turkey
Email: gedik@ybu.edu.tr

A. Aydin Alatan, Senior Member, IEEE
Department of Electrical and Electronics Engineering,
Middle East Technical University,
Ankara, Turkey
Email: alatan@eee.metu.edu.tr

***Abstract*—Performing high accurate pose estimation has been an attractive research area in the field of computer vision; hence, there are a plenty of algorithms proposed for this purpose. Starting with RGB or gray scale image data, methods utilizing data from 3D sensors, such as Time of Flight (TOF) or laser range finder, and later those based on RGBD data have emerged chronologically. Algorithms that exploit image data mainly rely on minimization of image plane error, i.e. the reprojection error. On the other hand, methods utilizing 3D measurements from depth sensors estimate object pose in order to minimize the Euclidean distance between these measurements. However, although errors in associated domains can be minimized effectively by such methods, the resultant pose estimates may not be of sufficient accuracy, when the dynamics of the object motion is ignored. At this point, the proposed 3D rigid pose estimation algorithm fuses measurements from vision (RGB) and depth sensors in a probabilistic manner using Extended Kalman Filter (EKF). It is shown that such a procedure increases pose estimation performance significantly compared to single sensor approaches.**

I. Introduction

Many computer vision related problems can be solved effectively with the aid of 3D object tracking. In robotic applications, knowing the exact metric location of entities relative to each other enables user defined actions to be performed by automatic systems. For instance, estimation of the relative orientation between Unmanned Aerial Vehicle (UAV) and tanker reference frames gains autonomous refueling capability to UAVs. Considering monetary as well as human health issues, highly accurate position estimation is critical for the reliability of such systems. As a military system, on May 2012, DARPA and Northrop Grumman Corp. announced completion of an autonomous refueling system between two high altitude UAVs [1].

Augmented Reality (AR) is also an attractive application of pose estimation. AR simply aims insertion of artificial objects to a real scene observed by a camera. To this aim, the relative orientation between capturing camera and the scene should be discovered. The challenge in such applications is to obtain consistent pose estimates at consecutive time instants in order to avoid jitter. In a practical system an automobile company aims to assist technical staff during car maintenance by insertion of synthetic information onto the observed motor video [2].

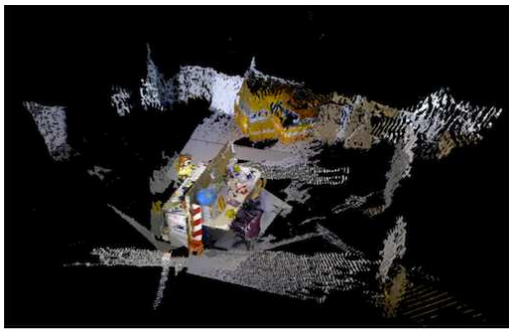


Fig. 1: 3D scene map [4]

In some applications, in which the object is represented as a point cloud model (PCM), estimation of orientation between camera and object reference frames enables 3D mapping of the object. For instance, the map shown in Figure 1 is obtained by tracking the object for more than 1000 frames and for such cases that may suffer from error accumulation over time, pose estimation accuracy becomes critical in order to generate pleasant 3D maps. There are commercial solutions that aim to bring such a technology into our homes [3].

Recent advances in sensor technology, created sensors, such as SwissRanger Sr-4000 [5], Microsoft Kinect [6] and Asus Xtion [7], that can capture high resolution and high frame rate 3D as well as 2D visual data. Among these sensors, Kinect and Xtion are widely used among computer vision researchers, since they are affordable as well as providing time synchronous RGB and 3D data. Figure 2 shows typical registered RGB and depth images that are captured by Kinect.

In this paper, our aim is to increase the accuracy of 3D pose estimation by simultaneous utilization of visual and depth data provided by the sensors. Figure 3 illustrates the object and sensor reference frames. Following variables define the overall system:

$\begin{bmatrix} X_{o_i} \\ Y_{o_i} \\ Z_{o_i} \end{bmatrix}$: 3D coordinates of i^{th} object point with respect to the object reference frame,

(a) RGB image (b) Depth image

Fig. 2: Typical RGB and depth images captured by Kinect [8].

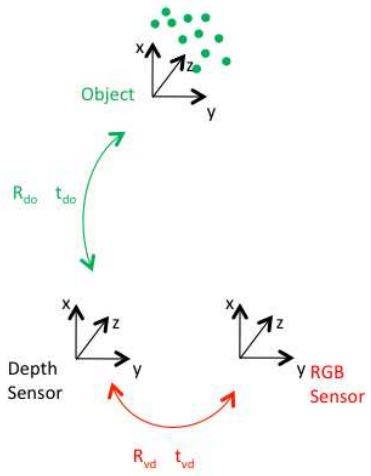


Fig. 3: Object and sensor reference frames.

$\begin{bmatrix} X_{o_{d_i}} \\ Y_{o_{d_i}} \\ Z_{o_{d_i}} \end{bmatrix}$: 3D coordinates of i^{th} object point measured by the depth camera with respect to the depth camera reference frame,

$\begin{bmatrix} x_{o_{v_i}} \\ y_{o_{v_i}} \end{bmatrix}$: 2D pixel coordinates of i^{th} object point measured by the vision sensor,

$R_{do} = R(\rho_{do}, \theta_{do}, \phi_{do})$: Rotation matrix between the object and the depth camera reference frames defined by angles ρ_{do}, θ_{do} and ϕ_{do} in x, y and z directions, respectively,

$t_{do} = [t_{x_{do}}, t_{y_{do}}, t_{z_{do}}]^T$: Translation vector between the object and the depth camera reference frames in x, y and z directions, respectively,

$R_{vd} = R(\rho_{vd}, \theta_{vd}, \phi_{vd})$: Rotation matrix between the depth and the vision sensor reference frames defined by angles ρ_{vd}, θ_{vd}, and ϕ_{vd}, in x, y and z directions, respectively,

$t_{vd} = [t_{x_{vd}}, t_{y_{vd}}, t_{z_{vd}}]^T$: Translation vector between the depth and the vision sensor reference frames in x, y and z directions, respectively.

Problem is defined as recovering the transformation between object and depth camera reference frames, i.e. R_{do} and t_{do}. We assume that external calibration between vision and depth sensors, i.e. R_{vd} and t_{vd}, is already estimated and remains fixed in time. Please refer to [9] for a discussion on external calibration of the sensors.

3D object coordinates and measurements of the depth sensor are related as follows (Time index is omitted for the sake of simplicity):

$$\begin{bmatrix} X_{o_{d_i}} \\ Y_{o_{d_i}} \\ Z_{o_{d_i}} \end{bmatrix} = R_{do} \begin{bmatrix} X_{o_i} \\ Y_{o_i} \\ Z_{o_i} \end{bmatrix} + t_{do} \tag{1}$$

Note that once 3D-3D correspondences between object coordinates and depth sensor measurements are known or estimated, it is possible to recover desired transformation parameters. Methods utilizing depth only measurements are examined in Section II.

Similarly, ignoring lens distortions [10] ,vision sensor measurements can be related to the object coordinates:

$$\alpha_i \begin{bmatrix} x_{o_{v_i}} \\ y_{o_{v_i}} \\ 1 \end{bmatrix} = K_v \left[R_{vd} \left[R_{do} \begin{bmatrix} X_{o_i} \\ Y_{o_i} \\ Z_{o_i} \end{bmatrix} + t_{do} \right] + t_{vd} \right] \tag{2}$$

where α_i is the scale factor and K_v is the internal calibration matrix of the vision sensor. K_v is related with the physical construction of the vision sensor and according to pin-hole camera model it can be written as follows [10]:

$$K_v = \begin{bmatrix} f_x & s & p_x \\ & f_y & p_y \\ & & 1 \end{bmatrix} \tag{3}$$

where f_x and f_y represent focal lengths, p_x and p_y represent principal point coordinates (all in pixels) and s is the skew parameter. Rewriting (2), we obtain the following relation for vision sensor measurements:

$$\alpha_i \begin{bmatrix} x_{o_{v_i}} \\ y_{o_{v_i}} \\ 1 \end{bmatrix} = K_v \left[R_{vo} \begin{bmatrix} X_{o_i} \\ Y_{o_i} \\ Z_{o_i} \end{bmatrix} + t_{vo} \right] \tag{4}$$

where

$$\begin{aligned} R_{vo} &= R_{vd} R_{do} \\ t_{vo} &= R_{vd} t_{do} + t_{vd} \end{aligned} \tag{5}$$

Similar to (1), once 2D-3D correspondences are established, it is possible to recover K_v, R_{vo} and t_{vo}, and hence, the required transformation $[R_{do}, t_{do}]$. In the literature, the estimation of K_v and $[R_{vo}, t_{vo}]$ from correspondences are referred as internal camera calibration and pose estimation, respectively. Although these two problems are coupled and can be solved simultaneously, throughout this paper, we assume that K_v is calculated offline and remains fixed. The algorithms proposed for pose estimation using 2D-3D correspondences are reviewed in Section II.

Although it is possible to estimate the object pose using single sensor approaches, in this paper we show that by fusing both sensor measurements in a probabilistic manner, it is possible to increase the accuracy of pose estimation. To this aim, EKF is adopted as the probabilistic framework due to its ability to model the dynamics of the object motion.

II. Related Work

For pose estimation using 2D-3D correspondences, many solutions have been proposed. Similar to the classification mentioned in [11], in a more general manner, these algorithms can be grouped into two categories as follows:

1) Algorithms directly utilizing 2D-3D correspondences to minimize errors, such as reprojection error, object space linearity error, etc.
2) Algorithms relying on the estimation of object coordinates with respect to the camera reference frame.

In one of the earliest approaches falling into the first category, a method named POSIT (Pose from Orthography and Scaling with Iterations) is proposed [12]. In POSIT, by updating the initial scale factor through the iterations, the perspective projection is estimated by scaled orthographic projection, until convergence. The extended version called SOFTPOSIT [13] solves the assignment, i.e. matching between 2D image and 3D object coordinates, and pose estimation problems simultaneously. The assignment problem is solved by the softassign algorithm of [14]. SOFTPOSIT algorithm is observed to be sensitive to initial conditions; hence, may diverge if not properly initialized.

The Direct Linear Transformation (DLT) [15] method estimates the 4×3 transformation matrix between 3D object coordinates and 2D image coordinates directly without forcing the rigid body transformation model and then the transformation matrix is decomposed into internal camera, rotation and translation matrices using RQ decomposition. Since this simple method is not accurate, it is generally utilized as an initial estimate in iterative approaches. However, due to inherent sensor noise, it is not trivial to decompose projection matrix in order to get an orthogonal rotation matrix and error is introduced during this decomposition. With this motivation, the algorithm proposed by the authors in [16] utilizes orthogonal iterations (OI) method, which guarantees an orthogonal rotation matrix R through iterations and decreases the object reference frame space error unless a solution is reached. The image points are utilized as hypothesized scene points in order to obtain an initial estimate and this initialization is stated to result with a pose estimation better than a weak-perspective initialization.

Finally, the author of [17] proposes an efficient linear solution for the exterior orientation estimation problem. Orthogonal decomposition is first used to isolate the unknown depths of feature points with respect to the camera reference frame. This approach allows the problem to be reduced to an absolute orientation with scale problem, which is solved using the Singular Value Decompositon (SVD).

The algorithms in the second category stem from the Perspective-n-Points (PnP) approach (specifically P3P approach) developed by Grunert in 1841 [18], which is still a highly popular method in pose estimation literature. The method utilizes the relative distances between the coordinates of features in object reference frame and provides a closed form solution to the corresponding 3D coordinates in the camera reference frame. Finally, the two reference frames are related by solving the absolute orientation problem. In [19], possible P3P solutions are analyzed.

The P3P method is quite sensitive to noise, since it depends on the solution of higher order polynomials for determination of the camera coordinates. Moreover, P3P algorithm may yield up to 4 real solutions. Hence, in [20], the authors propose a linear method, which is based on solving many P3P equations from n ($n > 3$) points, using SVD. However, this approach is not stable when there are outliers and a positive solution is not guaranteed. Therefore, the authors of [21] propose a PnP solution utilizing Gauss-Newton iterations based on the initial estimates of [20]. The iterated manner, however, may decrease efficiency. An efficient non-iterative PnP algorithm, which is based on expressing the 3D feature points as a weighted sum of four virtual control points and estimating the coordinates of these control points in the camera reference frame, is proposed in [22]. The method is stated to be efficient with $O(n)$ complexity.

Moreover, although their accuracies are generally lower than mentioned methods, there are numerous algorithms utilizing pattern recognition techniques [23] in the literature of pose estimation. For instance, the head pose estimation algorithm proposed in [24] utilizes spectral regression discriminant analysis with automatic regularization parameter estimation. The method is claimed to yield promising pose estimation results. There are also multi-view approaches proposed for pose estimation. In [25], data fusion is performed by back-projections from single images of the multi-view set onto the estimated 3D model. Then, the model pan angle is estimated by utilizing a particle filter. Furthermore, in [26], a neural network-based multi-view pose estimation scheme is proposed.

The algorithms utilizing pure range data are based on the registration of 3D point clouds, especially when the 3D-3D correspondences are not known. For instance in [27], in offline phase, a triangular mesh model of the object is formed using range sensors. In the online stage, the mesh model of the object is aligned with the captured range data using Iterative Closest Point (ICP) approach. Proposed in [28], ICP algorithm first matches the closest points between two point sets and calculates pose based on this association. Then aligns points using this pose estimate and in an iterative manner continues to match nearest points and estimate pose. For pose estimation, quaternion-based approach, which represents rotation matrix using a 4×1 unit norm vector, is used. The quaternions are calculated by minimizing a mean square objective function that calculates the difference between original and transformed 3D points. The authors of [27] modify ICP so that it can be implemented in real-time. The main drawback of ICP algorithm is its sensitivity to initial object pose. Since ICP requires a good initial estimate to converge, many algorithms exploit ICP after a coarse registration. In [29], using the limits of object velocity and the sensor frame rate, the interframe transformation space is reduced considerable and the pose space is quantized; hence the problem of pose estimation is converted to a classification problem. Following discrete

pose classification step, ICP algorithm is utilized to fine-tune pose estimates with a few iterations. Furthermore, utilizing parallel processing on GPUs, in [30] an ICP based 3D tracking algorithm is proposed. The system tracks all pixels in a 640x480 image and the performance is quite satisfactory due to dense tracking. However, the huge computational burden makes it impossible to run the algorithm on conventional desktop platforms.

In order to perform 3D-3D registration a descriptor based approach is proposed in [31]. First of all, for each data point, a descriptor based on local geometry is computed. Then distinctive features are selected among data points based on the uniqueness of their descriptors. Then, a distance matrix storing the descriptor distances between model and data points is formed. Optimal set of correspondences, which brings the sets to a coarse alignment, is established using the branch-and-bound algorithm. The pose is refined further using ICP method. However, in the case of noisy measurements, a descriptor-based method may not yield satisfactory results.

One of the main drawbacks of working with range data is the inherent sensor noise and low resolution. Hence, in order to overcome such limitations, many researches propose to utilize range sensors and vision sensors together. In [32], the 2D-3D correspondences between range and vision data are utilized for RANSAC [33] and LM [34] based pose estimation. Sensor fusion is only utilized to calculate covariance matrices of 3D measurements and pose tracking is instantaneous. To handle such drawbacks, in [35], first the object is transformed using the initial state estimate of EKF. Then, the proposed articulated ICP is used to align point clouds. Finally, measurement update is performed to correct the pose estimate of the articulated ICP. Moreover, the pose estimates of EKF and ICP based methods *corrects* each other at each iteration. Similarly, the authors of [36] maximize photo consistency by linearizing the cost function to register all pixels of consecutive frames. The authors of [37] propose a probabilistic optimization framework in order to register RGBD images and track pose. The joint shape and color distributions are represented as a tree structure, where each node stores statistics on the joint spatial and color distributions of the points within its volume. The graphs at different time instants are associated by finding the transformation (represented by unit quaternions and a translation vector) that maximizes their matching likelihood. Although method is accurate, it is computationally involved.

Although not pointed by the above algorithms, the motion of a free moving object can be modeled using a dynamic system framework, such that the current pose of the object is constrained with the previous time instant pose and the underlying *motion model*. This formulation enables time consistent tracking results; hence, reduces jitter. By this motivation, the algorithm of [35] performs a weighted combination of object pose imposed by motion model and that estimated using an ICP variant operating on RGBD images. However, in this approach a higher level fusion of estimated pose parameters is proposed and underlying noise statistics inherent in the measurements of vision and depth sensors is ignored.

III. Probabilistic Fusion of RGBD Data

As illustrated in Figure 3, the joint utilization of vision and depth sensors enables two sets of measurements for each object point *i*, namely $[X_{o_{d_i}}, Y_{o_{d_i}}, Z_{o_{d_i}}]^T$ from depth sensor and $[x_{o_{v_i}}, y_{o_{v_i}}]^T$ from vision sensor. In order to increase the accuracy of 3D pose estimation, one should devise an algorithm exploiting these two types of data as much as possible. Researchers involved in probabilistic robotics research mainly deal with solving similar problems. The robot needs to make an estimate of the *state* using *measurements* acquired by its sensors. The term *state* stands for any property of the robot or the environment of interest, such as the velocity or position of the robot and the locations of features around the robot [38]. The states change in time according to a system model. For instance, if you apply *this* much power to the wheel motors, the velocity of the robot changes *that* much. These states cannot be directly observed by the robot but through some sensor *measurements*, such as the readings of an odometer sensor or the images captured by a vision sensor. Similarly, these measurements are mathematically related with the states. To sum up, two relations govern the overall state estimation procedure:

1) **State Update Equations:** What is the state x_t at time t, if a sequence of control inputs $u_{1:t}$ are applied, a sequence of states $x_{0:t-1}$ are resulted and a sequence of measurements $z_{1:t-1}$ are observed?
2) **Measurement Equations:** What is the measurement z_t at time t, if a sequence of control inputs $u_{1:t}$ are applied, a sequence of states $x_{0:t}$ are resulted and a sequence of measurements $z_{1:t-1}$ are observed?

States and measurements evolve in time according to probabilistic laws [38]. State and measurement equations are governed by following probability distributions, respectively:

$$\begin{gathered} p(x_t|x_{0:t-1}, z_{1:t-1}, u_{1:t}) \\ p(z_t|x_{0:t}, z_{1:t-1}, u_{1:t}) \end{gathered} \tag{6}$$

Although the robot cannot directly observe states, it can have a *belief* regarding the states through measurement and control input sequences:

$$bel(x_t) = p(x_t|z_{1:t}, u_{1:t}) \tag{7}$$

Defined formally, belief is the robot's internal knowledge of the state. A two step approach gives the belief. First, before utilizing the current measurement z_t, a *prediction* $\overline{bel}(x_t)$ is made:

$$\overline{bel}(x_t) = p(x_t|z_{1:t-1}, u_{1:t}) \tag{8}$$

Then after incorporating z_t, $bel(x_t)$ is obtained from $\overline{bel}(x_t)$ by *measurement update*. *Bayes filter* algorithm is the most general method for calculating beliefs [38]. In a Markov process, the prediction and measurement update steps of the algorithm are respectively as follows:

$$\begin{aligned} \overline{bel}(x_t) &= \int p(x_t|x_{t-1}, u_t) bel(x_{t-1}) dx \\ bel(x_t) &= \eta p(z_t|x_t) \overline{bel}(x_t) \end{aligned} \tag{9}$$

where η is the normalizing scalar. Please note that, once the dynamical system model, defined by state update and measurement equations, is known, it is possible to obtain the probabilities $p(x_t|x_{t-1}, u_t)$ and $p(z_t|x_t)$; hence, the belief, using Bayes filter approach. For the 3D pose estimation problem of concern, in which the object makes free movements and associated measurements are acquired by sensors, Bayes algorithm stands as a powerful tool to estimate pose, since it makes time consistent estimations in a well-defined probabilistic framework. The mathematical derivations of Bayes algorithm are not in the scope of this manuscript. Refer to [38] for further discussions.

Kalman Filter (KF) is proposed for the solution of Bayes algorithm in linear systems, for which the state transition ($p(x_t|x_{t-1}, u_t)$) and measurement ($p(z_t|x_t)$) probabilities are linear in terms of their arguments with additive Gaussian noise [39]. However, as introduced in Section I, geometric relations involved in model-based 3D tracking have non-linear characteristics. Therefore, KF formulation does not directly suit our needs. At this point, Extended Kalman Filter (EKF) proposed for non-linear systems comes as a solution, for which state update and measurement equations are governed by functions *g* and *h*, respectively:

$$\begin{aligned} x_t &= g(u_t, x_{t-1}) + \epsilon_t \\ z_t &= h(x_t) + \varepsilon_t \end{aligned} \tag{10}$$

where ϵ_t and ε_t are zero mean random Gaussian vectors standing for randomness in state transition and measurements with covariance matrices R_t and Q_t respectively. Consequently, the EKF algorithm estimates the belief as follows [38]:

$$\begin{aligned} \overline{\mu}_t &= g(u_t, \mu_{t-1}) \\ \overline{\Sigma}_t &= G_t \Sigma_{t-1} G_t^T + R_t \\ K_t &= \overline{\Sigma}_t H_t^T (H_t \overline{\Sigma}_t H_t^T + Q_t)^{-1} \\ \mu_t &= \overline{\mu}_t + K_t (z_t - h(\overline{\mu}_t)) \\ \Sigma_t &= (I - K_t H_t) \overline{\Sigma}_t \\ bel(x_t) &= N(x_t; \mu_t, \Sigma_t) \end{aligned} \tag{11}$$

where G_t and H_t stand for *Jacobians* and N is the Gaussian distribution. The underlying motion model of the probabilistic system defines the state update equation, and hence the transition between adjacent states. Although in robotic applications, there are a variety of motion models depending on the type of the robot and the kinematics of its moving parts, in our case of a single independently moving object, the underlying motion model is relatively easily defined. The object moves with respect to either constant *position*, constant *velocity* or constant *acceleration* model. In vision literature, constant velocity motion model is usually utilized to model motion of free moving hand-held cameras and rigid objects [40], [35], [41], [42]. In constant velocity motion model, the state is composed of position and velocity of the object. The velocity between consecutive time instants is the same up to an additive noise term. This noise term accounts for any possible acceleration caused by system dynamics or external influences. On the other hand, the position is updated by simple addition of previous position, change in position (velocity times delta time) and a noise term.

The aim of the Bayes filter (actually the EKF) is to estimate the states, i.e. R_{do}, t_{do} and associated velocities, by using the observations $[X_{o_{d_i}}, Y_{o_{d_i}}, Z_{o_{d_i}}]^T$ and $[x_{o_{v_i}}, y_{o_{v_i}}]^T$. In constant velocity motion model, the state update equations, relating consecutive states, can be written as follows:

$$\begin{aligned} \begin{bmatrix} \rho_{do} \\ \theta_{do} \\ \phi_{do} \\ t_{x_{do}} \\ t_{y_{do}} \\ t_{z_{do}} \end{bmatrix}_t &= \begin{bmatrix} \rho_{do} \\ \theta_{do} \\ \phi_{do} \\ t_{x_{do}} \\ t_{y_{do}} \\ t_{z_{do}} \end{bmatrix}_{t-1} + \begin{bmatrix} \dot{\rho}_{do} \\ \dot{\theta}_{do} \\ \dot{\phi}_{do} \\ \dot{t}_{x_{do}} \\ \dot{t}_{y_{do}} \\ \dot{t}_{z_{do}} \end{bmatrix}_{t-1} \times \Delta t + \epsilon_t^i \\ \begin{bmatrix} \dot{\rho}_{do} \\ \dot{\theta}_{do} \\ \dot{\phi}_{do} \\ \dot{t}_{x_{do}} \\ \dot{t}_{y_{do}} \\ \dot{t}_{z_{do}} \end{bmatrix}_t &= \begin{bmatrix} \dot{\rho}_{do} \\ \dot{\theta}_{do} \\ \dot{\phi}_{do} \\ \dot{t}_{x_{do}} \\ \dot{t}_{y_{do}} \\ \dot{t}_{z_{do}} \end{bmatrix}_{t-1} + \epsilon_t^{ii} \end{aligned} \tag{12}$$

The first line of (12) performs position update by adding previous position and position update, whereas the second line stands for the conservation of velocity up to an additive noise term. It should be noted that time difference (Δt) between consecutive updates is 1 frames.

On the other hand, measurement equations relating current states and current measurements are exactly same as (1) and (2), except an additive noise term to account for the measurement noise. For N object points, $5N \times 1$ measurements are obtained by concatenating each sensor measurements:

$$\begin{aligned} \begin{bmatrix} X_{o_{d_i}} \\ Y_{o_{d_i}} \\ Z_{o_{d_i}} \end{bmatrix}_t &= [R_{do}]_t \begin{bmatrix} X_{o_i} \\ Y_{o_i} \\ Z_{o_i} \end{bmatrix} + [t_{do}]_t + \varepsilon_t^i \\ \alpha_i \begin{bmatrix} x_{o_{v_i}} \\ y_{o_{v_i}} \\ 1 \end{bmatrix}_t &= K_v \left[R_{vd} \left[[R_{do}]_t \begin{bmatrix} X_{o_i} \\ Y_{o_i} \\ Z_{o_i} \end{bmatrix} + [t_{do}]_t \right] + t_{vd} \right] + \varepsilon_t^{ii} \end{aligned} \tag{13}$$

where $[X_{o_i}, Y_{o_i}, Z_{o_i}]^T$ represents 3D coordinates of i^{th} object point with respect to object reference frame and subscript t denotes time. As a final remark, instead of a $5N \times 1$ measurement vector, a possible combination that yields a 5×1 measurement vector could be considered. In [43], it is shown for the linear Kalman case that the former approach is more flexible and computationally more efficient for time-varying noise characteristics and increased number of measurements. Although, the noise on measurements are highly dependent on used vision and depth sensors, the scene characteristics and the feature matching algorithm used, without loss of generality,

the measurement noise covariance matrix Q_t is designed in the form of a diagonal matrix of size $5N \times 5N$ where each entry specifies variance of associated measurement:

$$Q_t = diag(\sigma^2_{XYZ}, \sigma^2_{pix}) \tag{14}$$

A general discussion on the noise model for the Kinect sensor can be found in [44]. Finally, the measurements can be associated between consecutive time instants using the feature tracking algorithm in [4].

IV. Test Results

The performance of the proposed RGBD data based pose estimation algorithm is compared to that of methods utilizing single sensor approaches. Please remember that, as detailed in Section I, the method using 3D measurements estimates object pose using (1). To this aim quaternion based approach [45], which is widely used in the associated literature, is adopted. The algorithm minimizes the 3D error that is the difference between transformed object coordinates and 3D measurements by the depth sensor $[X_{o_{d_i}}, Y_{o_{d_i}}, Z_{o_{d_i}}]^T$:

$$e_{3D} = \| \begin{bmatrix} X_{o_{d_i}} \\ Y_{o_{d_i}} \\ Z_{o_{d_i}} \end{bmatrix}_t - \left[[R_{do}]_t \begin{bmatrix} X_{o_i} \\ Y_{o_i} \\ Z_{o_i} \end{bmatrix} + [t_{do}]_t \right] \| \tag{15}$$

On the other hand, the approach depending on 2D measurements to estimate pose uses (2) and minimizes the reprojection error between pixel measurements $[x_{o_{v_i}}, y_{o_{v_i}}]^T$ and object coordinates $[X_{o_i}, Y_{o_i}, Z_{o_i}]^T$ projected on the image plane:

$$e_{2D} = \| \begin{bmatrix} x_{o_{v_i}} \\ y_{o_{v_i}} \end{bmatrix}_t - \begin{bmatrix} x_{o_{v_i}} \\ y_{o_{v_i}} \end{bmatrix}_{t,P} \| \tag{16}$$

The approach finds initial pose estimate using PnP algorithm [21] and refines it further using LM optimization [34]. Note that these single sensor algorithms are selected based on the observation that many end-to-end 3D trackers utilize these methods as the core pose estimation routine.

In order to analyze the performance of the methods, an artificial test scenario is designed. To this aim, the *Face* data set, with the 3D model shown in Figure 4, is used. Since the model is composed of thousands of points, random 20 points are selected as $[X_{o_i}, Y_{o_i}, Z_{o_i}]^T$ and utilized for tracking.

The initial states at time t_0 are selected as follows:

$$\begin{bmatrix} \rho_{do} \\ \theta_{do} \\ \phi_{do} \\ t_{x_{do}} \\ t_{y_{do}} \\ t_{z_{do}} \end{bmatrix}_{t_0} = \begin{bmatrix} 1 \times 10^{-4} \quad rad \\ -1.4 \quad rad \\ 1 \times 10^{-4} \quad rad \\ 50 \quad mm \\ 50 \quad mm \\ 2000 \quad mm \end{bmatrix}$$
$$\begin{bmatrix} \dot{\rho}_{do} \\ \dot{\theta}_{do} \\ \dot{\phi}_{do} \\ \dot{t}_{x_{do}} \\ \dot{t}_{y_{do}} \\ \dot{t}_{z_{do}} \end{bmatrix}_{t_0} = \begin{bmatrix} 1 \times 10^{-4} \quad rad/frame \\ 25 \times 10^{-3} \quad rad/frame \\ 1 \times 10^{-4} \quad rad/frame \\ 1 \times 10^{-4} \quad mm/frame \\ 1 \times 10^{-4} \quad mm/frame \\ 1 \times 10^{-4} \quad mm/frame \end{bmatrix} \tag{17}$$

Fig. 4: 3D model of the *Face* sequence.

The states at consecutive time instants are obtained according to (12). Similarly, (13) is used to generate the measurements $[X_{o_{d_i}}, Y_{o_{d_i}}, Z_{o_{d_i}}]^T$ and $[x_{o_{v_i}}, y_{o_{v_i}}]^T$ at each time instant for the points selected from the model. The measurement noise variances σ^2_{XYZ} and σ^2_{pix} are 10 mm and 0.5 pixels for 3D and 2D measurements, respectively. The sequence consists of 100 frames, therefore, we simulate a movement of the head from left to right with a dominant motion in the y-axis. The results are obtained by Monte Carlo simulations composed of 50 trials. The sensors are calibrated internally and externally by using the procedure detailed in [9]. Moreover, the initial state is assumed to be known for all methods.

Proposed method and single sensor approaches are compared in terms of 2D reprojection error (16), 3D error (15) and deviation from groundtruth pose parameters available. Mean error values are obtained by averaging associated values for all tracked object points in a frame. 2D reprojection errors for methods are shown in Table I.

TABLE I: Mean reprojection errors (in pixels).

Method	**Fusion**	**2D only**	**3D only**
Error	0.88	**0.82**	0.92

By construction, the algorithm utilizing vision sensor minimizes the reprojection error. Therefore, in terms of reprojection error it gives the best performance. On the other hand, 3D errors are obtained as shown in Table II.

TABLE II: Mean 3D errors (in mm).

Method	**Fusion**	**2D only**	**3D only**
Error	4.96	30.72	**4.77**

Similarly, method utilizing mere depth sensor measurements minimizes 3D error; and hence it gives best performance in terms of this metric. Finally, mean pose estimation error values are tabulated in Table III. The sensor fusion approach minimizes 2D and 3D errors simultaneously and performs more accurate pose estimation. Therefore, the proposed fusion-based method is much more reliable compared to single sensor based algorithms.

TABLE III: Mean tracking errors.

Method	Fusion	2D only	3D only
rotation-x (mrad)	**6.14**	48.34	23.76
rotation-y (mrad)	**2.83**	35.48	15.96
rotation-z (mrad)	**6.45**	28.48	22.61
translation-x (mm)	**1.60**	29.37	10.98
translation-y (mm)	**2.73**	27.77	9.38
translation-z (mm)	**1.71**	29.16	8.40

Instead of PnP and quaternion based algorithms, EKF is also utilized to perform pose estimation using single sensor inputs. The state update equation is exactly same as (12), however $2N \times 1$ and $3N \times 1$ measurement vectors for methods utilizing 2D-3D and 3D-3D correspondences are obtained using the first and second lines of (13) respectively. The pose estimation accuracies are as follows:

TABLE IV: Mean tracking errors for single sensor methods.

Method	2D only EKF	3D only EKF
rotation-x (mrad)	15.11	7.08
rotation-y (mrad)	9.39	3.01
rotation-z (mrad)	10.96	8.41
translation-x (mm)	6.92	1.68
translation-y (mm)	7.84	2.76
translation-z (mm)	7.70	1.79

It is clear from Table IV that, although the methods are much more accurate than their instantaneous counterparts depicted in Table III, they cannot perform better that the proposed sensor fusion approach.

Figure 5 illustrates the pose estimation errors and associated variance estimates by the filter (please note that only values for dominant y-directional rotation estimate is given for space restrictions). As the filter is updated error values converges towards zero, also the variance of the estimate is decreases, as expected. Therefore, we can conclude that variances can be utilized as a figure of merit for deduction of qualities of estimates. Finally, it is observed that the filter can tolerate up to 5% error during initialization and cannot converge after that point.

V. Conclusion

Depth and vision sensors provide data having completely different statistics and an optimal tracking method should handle this issue by considering the underlying noise models. Moreover, in 3D object pose estimation, the pose estimates need to be temporally consistent in order to reduce jitter and generate visually more pleasant tracks. Considering these preconditions, EKF comes as a powerful solution that can fuse RGBD data in order to solve nonlinear pose estimation problem. Hence, in this paper, starting from the basic state update and measurement equations, the proposed sensor fusion approach that adopts constant velocity motion model is detailed. The performance of the proposed formulation is analyzed in terms accuracy compared to single sensor approaches. It is observed that although single sensor approaches successfully minimize errors in 2D and 3D spaces, their accuracies are much lower than the sensor fusion approach.

Acknowledgement

This work is partially supported by The Scientific and Technological Research Council of Turkey. O. Serdar Gedik was at Middle East Technical University Department of Electrical and Electronics Engineering at the time of this work.

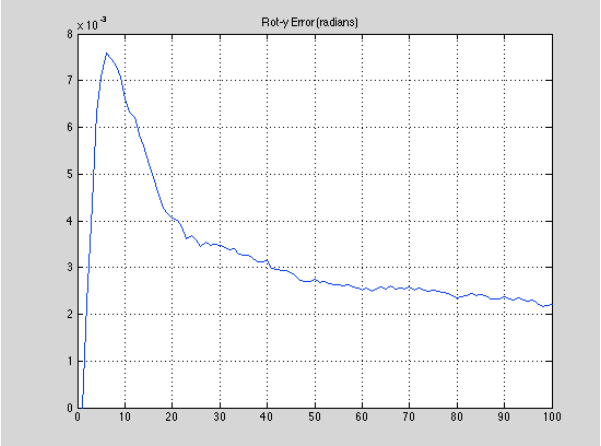
References

[1] NASA, "Nasa global hawks aid uav-to-uav refueling project," http://www.nasa.gov/centers/dryden/statusreports/globalhawkstatus100512.html, accessed: 07/04/2014.

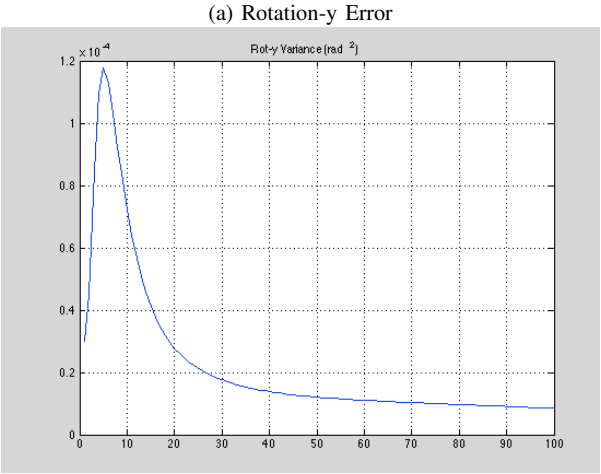
[2] BMW, "Bmw augmented reality," http://www.bmw.com/com/en/owners/service/, accessed: 07/04/2014.

[3] MatterPort, "Matterport 3d models for real interior spaces," http://matterport.com/, accessed: 07/04/2014.

[4] O. S. Gedik and A. A. Alatan, "3d rigid body tracking using vision and depth sensors," *IEEE Transactions on Cybernetics Part B*, vol. 43, no. 5, pp. 1395–1405, 2013.



(a) Rotation-y Error



(b) Rotation-y Variance

Fig. 5: Error and variance values.

[5] MESA, "Mesa imaging," http://www.mesa-imaging.ch/products/product-overview/, accessed: 07/04/2014.

[6] Microsoft, "Microsoft kinect," http://www.xbox.com/en-GB/kinect, accessed: 04/04/2013.

[7] ASUS, "Multimedia xtion pro," http://www.asus.com/Multimedia/XtionPRO/, accessed: 07/04/2014.

[8] J. Sturm, S. Magnenat, N. Engelhard, F. Pomerleau, F. Colas, W. Burgard, D. Cremers, and R. Siegwart, "Towards a benchmark for rgb-d slam evaluation," in *RGB-D Workshop on Advanced Reasoning with Depth Cameras*, 2011.

[9] O. S. Gedik and A. A. Alatan, "Fusing 2d and 3d clues for 3d tracking using visual and range data," in *The International Conference on Information Fusion*, 2013.

[10] R. Hartley and A. Zisserman, *Multiple View Geometry in Computer Vision*. Cambridge University Press, 2003.

[11] J. Lima, F. Simes, L. Figueiredo, V. Teichrieb, and J. Kelner, "Online monocular markerless 3d tracking for augmented reality," in *Abordagens Prticas de Realidade Virtual e Aumentada: SVR*, 2009.

[12] D. F. DeMenthon and L. S. Davis, "Model-based object pose in 25 lines of code," *International Journal of Computer Vision*, vol. 15, no. 1–2, pp. 123–141, 1995.

[13] P. David, D. F. DeMenthon, R. Duraiswami, and H. Samet, "Softposit: Simultaneous pose and correspondence determination," in *European Conference on Computer Vision*. Springer, 2002.

[14] S. Gold and S. Rangarajan, "A graduated assignment algorithm for graph matching," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 18, no. 4, pp. 377–388, 1996.

[15] R. I. Hartley, "Minimizing algebraic error in geometric estimation problems," in *IEEE International Conference on Computer Vision*, 1998.

[16] C. Lu and G. Hager, "Fast and globally convergent pose estimation from video images," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 22, no. 6, pp. 610–622, 2000.

[17] P. D. Fiore, "Efficient linear solution of exterior orientation," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 23, no. 2, pp. 140–148, 2001.

[18] A. J. Grunert, "Das pothenotische problem in erweiterter gestalt nebst bber seine anwendungen in der geodasie," in *Grunerts Archiv fur Matheinatik und Physik Band 1*, 1841.

[19] R. M. Haralick, C. Lee, and M. Nolle, "Analysis and solutions of the three point perspective pose estimation problem," in *IEEE Conference on Computer Vision and Pattern Recognition*, 1991.

[20] L. Quan and Z. Lan, "Linear n-point camera pose determination," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 21, no. 8, pp. 774–780, 1999.

[21] S. Ohayon and E. Rivlin, "Robust 3d head tracking using camera pose estimation," in *IEEE International Conference on Pattern Recognition*, 2006.

[22] F. Moreno-Noguer, L. V., and P. Fua, "Accurate non-iterative o(n) solution to the pnp problem," in *IEEE International Conference on Computer Vision*, 2007.

[23] E. Murphy-Chutorian and M. M. Trivedi, "Head pose estimation in computer vision: A survey," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 31, no. 4, pp. 607–626, 2009.

[24] C. Shan and W. Chen, "Head pose estimation using spectral regression discriminant analysis," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2009.

[25] C. Canton-Ferrer, C. J. R., and M. Pardas, "Head orientation estimation using particle filtering in multiview scenarios," in *Multimodal Technologies for Perception of Humans*. Springer, 2008.

[26] M. Voit, K. Nickel, and R. Stiefelhagen, "Neural network-based head pose estimation and multi-view fusion," in *Multimodal Technologies for Perception of Humans*. Springer, 2007.

[27] D. A. Simon, M. Hebert, and T. Kanade, "Real-time 3-d pose estimation using a high-speed range sensor," in *IEEE Conference on Robotics and Automation*, 1994.

[28] P. J. Besl and N. D. McKay, "A method for registration of 3d shapes," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 14, no. 2239–256, 1992.

[29] L. Shang, B. Jasiobedzki, and M. Greenspan, "Model-based tracking by classification in a tiny discrete pose space," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 29, no. 6, pp. 976–989, 2007.

[30] S. Izadi, D. Kim, O. Hilliges, D. Molyneaux, R. Newcombe, P. Kohli, J. Shotton, S. Hodges, D. Freeman, A. Davison, and A. Fitzgibbon, "Kinectfusion: Real-time 3d reconstruction and interaction using a moving depth camera," in *ACM Symposium on User Interface Software and Technology*, 201.

[31] N. Gelfand, N. J. Mitra, L. J. Guibas, and H. Pottmann, "Robust global registration," in *Eurographics Symposium on Geometry Processing*, 2005.

[32] B. Streckel, B. Bartczak, R. Koch, and A. Kolb, "Supporting structure from motion with a 3d-range-camera," *Lecture Notes in Computer Science*, vol. 4522, pp. 233–242, 2007.

[33] M. A. Fischler and B. C. Bolles, "Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography," *Communications of the ACM*, vol. 24, no. 6, pp. 381–395, 1981.

[34] W. H. Press, S. A. Teukolsky, W. T. Vetterling, and B. P. Flannery, *Numerical Recipes in C - The Art of Scientific Computing*. Cambridge University Press, 1992.

[35] M. Krainin, P. Henry, X. Ren, and D. Fox, "Manipulator and object tracking for in-hand 3d object modeling," *The International Journal of Robotics Research*, vol. 30, no. 11, pp. 1311–1327, 2011.

[36] F. Steinbrcker, J. Sturm, and D. Cremers, "Real-time visual odometry from dense rgb-d images," in *IEEE International Conference on Computer Vision Workshops*, 2011.

[37] J. Stckler and S. Behnke, "Model learning and real-time tracking using multi-resolution surfel maps," in *Twenty Sixth AAAI Conference on Artificial Intelligence*, 2012.

[38] S. Thrun, W. Burgard, and D. Fox, *Probabilistic Robotics*. MIT Press, 2005.

[39] R. E. Kalman, "A new approach to linear filtering and prediction problems," *Transactions of the ASME–Journal of Basic Engineering*, vol. 82, no. Series D, pp. 35–45, 1960.

[40] A. Davison, I. D. Reid, M. N. D, and O. Stasse, "Monoslam: Real-time single camera slam," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 29, no. 6, pp. 1052–1067, 2007.

[41] K. Nickels and S. Hutchinson, "Weighting observations: The use of kinematic models in object tracking," in *IEEE International Conference on Robotics and Automation*, 1998.

[42] G. Taylor and L. Kleeman, "Fusion of multimodal visual cues for model-based object tracking," in *Australasian Conference on Robotics and Automation*, 2003.

[43] Q. Gan and C. J. Harris, "Comparison of two measurement fusion methods for kalman-filter-based multisensor data fusion," *IEEE Transactions on Aerospace and Electronic Systems*, vol. 37, no. 1, pp. 273–279, 2001.

[44] C. V. Nguyen, S. Izadi, and D. Lovell, "Modeling kinect sensor noise for improved 3d reconstruction and tracking," in *International Conference on 3D Imaging, Modeling, Processing, Visualization and Transmission*, 2012.

[45] R. Jain, R. Kasturi, and B. G. Schunck, *Machine Vision*. McGraw-Hill, 1995.