ORB-SLAM AND INERTIAL SENSOR FUSION FOR GPS DENIED ENVIRONMENTS FOR AUTONOMOUS CAR LOCALIZATION

H. Burak KURT, Erdinç ALTUĞ

[1,2] Istanbul Technical University, Faculty of Mechanical Engineering, Istanbul, Turkey
[2] Corresponding Author: Erdinç Altuğ, altuger@itu.edu.tr

ABSTRACT

The level of autonomy of autonomous vehicles has increasing considerably within the last couple of years. Nowadays cars are expected to work on complex, unknown environments effectively. These days, there are many studies autonomously driven car for outdoor environment. These cars have very reliable localization and mapping performance using GPS. On the other hand, GPS does not work reliable for indoor environment. In this paper, GPS denied indoor environment localization is studied. We propose combination of visual and inertial sensors for localization. We have implemented the ORB-SLAM algorithm computer. After that IMU and ORB-SLAM algorithm are combined for finding optimum localization data. In order to combine these two sensors, linear kalman filter is used. All of implementation and filtering of two sensors is done using ROS (Robot Operating System). This system was installed on RC car to demonstrate the effectiveness of the system. Experiments performed on indoor environment show the advantages and limitations of the system.

Keywords: ORB-SLAM, Kalman filter, Localization, ROS, Autonomous vehicles.

1. INTRODUCTION AND MAIN SECTIONS

Nowadays autonomous cars are expected to drive autonomously on complex environments such as crowded indoor environments or busy streets. The successful operation of a car depends heavily on knowing the exact location of the car throughout the motion, which is known as localization. If the map of the environment is known, localization is simply determining where the car is on this map. Localization is very important in autonomous vehicle area. When there is no map available, then localization as well as map building will be necessary at the unknown environment to autonomously drive the car.

For car localization there are different techniques; odometry, Global Navigation Satellite System (GNSS), inertial navigation and visual techniques. For outdoor environments GNSS information obtained from orbiting navigation satellites provide this information freely. For indoor environments (and at some outdoor areas as well) relying on GNSS is not an option due the fact that the GPS data is not reliable or non-existing due the inability to access enough satellite data.

Another approach on localization is to use vehicle tire rotation as well as steering wheel angle. This approach is called odometry. This approach is possible to use for car localization. However there are some drawbacks of this approach. It suffers from position drift and inaccuracy because of tire slippage. If tire of car slips, measurement of rotation of tire gives wrong information. As result of this situation localization of car is not reliable. Also translation and orientation errors in tire odometry increase with time and total travelled distance have big error.

One other approach is the use of an inertial navigation system (INS), which relies on inertial sensory techniques to estimate vehicle position. In this approach Micro Electro Mechanical systems (MEMS) based micro sensors involving accelerometers, gyros, and magnetometer are combined in a package, and a microprocessor fuses the data through a filter to calculate the vehicle position. If GNSS data is available, the estimated of INS can be updated or calibrated. This approach is called as GNSS/INS integration, and it proposes even smaller errors. Xsens MTI-G-700, proposes accuracy of 2 meters depending on GNSS availability and placement of the antenna (URL 1). Although GNSS based techniques are simple, easy to implement and proposes small localization errors, it is not applicable to our indoor environments due to no GNSS reception.

Localization of autonomous cars can also be done using visual information obtained through cameras. Some techniques use artificial visual clues placed on the environment. 2D fiducial markers are popular for localization

due to fast and robust behavior. Placing unique markers to the know locations of the environment help the vehicle to localize itself. The markers should be in acceptable size, to be seen by the camera in a distance, but not too big to disturb the environment. They should be detected in real-time with a computer. They should be robust to lighting condition changes, which is one of the most limiting factors in successful vision systems in real-world conditions. Moreover, they have to robust on blurred marker detection which occurs as the camera moves, as well robust on detecting partially missing or occluded markers that may be placed on the environment. Comparison of some of these techniques is presented in (Garrido-Jurado continued, 2016), (Muñoz-Salinas continued, 2016).

Placing artificial markers or patterns on an environment for localization is not very practical. Localization of autonomous car simultaneously, as it moves using vision and tracking only natural features (such as edges) is called as, simultaneous localization and map building (SLAM). One of the first feature-based real-time SLAM approach was proposed by Klein and Murray (Klein and Murray, 2007) as the PTAM method. It involves concurrent tracking and mapping of features. Davidson (Davison continued, 2007) used the SFM reconstruction technique for parallel tracking and mapping. Bora and Erdinç used this technique on the localization and stabilization of a quadrotor (Yiğit and Altuğ, 2012).

More recently, another SLAM approach was proposed that is based on extraction of ORB features (Mur-Artal continued, 2015). This approach is called as ORB-SLAM and it is shown to be more robust and more suitable for autonomous car application. It is shown to operate in real time, in small and large, indoor and outdoor environments. The accuracy of the system was reported to be in the order of cm in indoor environments and up to a few meters in large outdoor environments. ORB-SLAM is very promising on autonomous vehicle. ORB-SLAM algorithm requires high computation power. The computer used on paper (Mur-Artal continued, 2015) was having an i7 processor. High computation power is not problem for big and large autonomous vehicle platform. However it is not effective, when vehicle platform is small. Some researches (Martinez-Carranza continued, 2015) preferred to transfer the onboard images to a ground computer for processing and implemented the ORB-SLAM. This approach limits the autonomy considerably and not feasible because of the limited range and connection problems involved with transmitting images and receiving data from ground computer. In this study mono-camera is used for ORB-SLAM algorithm. Yingcai (Bi continued, 2016) propose the use of ORB-SLAM on a MAV with a dual-camera system.

Sensor fusion algorithm is commonly used for combining sensors. GPS and inertial sensor fusion application is very effective. However it can only work outdoor environment. Visual and inertial sensor fusion algorithms are can be used for indoor environment. Chang (Liu continued, 2016) propose semi direct monocular visual odometry (SVO) and freeIMU sensor fusion. In this study orientation fusion are done gradient descent optimization methods. ORB-SLAM algorithm is more accurate than SVO visual odometry algorithm and our application ORBSLAM is used and linear kalman filter is used instead of gradient descent optimization method.

In this paper, we propose an onboard vehicle localization for autonomous car platform. Visual and inertial sensor fusion is used to localize vehicle. Sensor fusion algorithm is separated two part, position fusion and orientation fusion. High computational computer (i7) is used for ORB-SLAM algorithm. Sensor fusion algorithm is used to obtain optimum localization measurement. Because ORB-SLAM algorithm is very accurate, but it is not speed enough. In situation of fast movement of vehicle ORB-SLAM algorithm performance decrease. On the other hand inertial sensor is not accurate like ORB-SLAM, but it measures data very fast. Fusion of visual and inertial sensor gives better localization measurement.

The rest of the article is organized as follows: section ‘localization and mapping’ gives ORB-SLAM algorithm; ‘Sensor fusion’ section gives overview of visual-inertial fusion algorithm; ‘Orientation fusion’ section gives mathematical model of orientation filter; ‘Position fusion’ section gives mathematical model of position filter; ‘Experiment’ section gives experiments are done; ‘ROS’ section express ROS (Robot Operating System); ‘Conclusion and recommendations’ section gives result of sensor fusion application and gives recommendations.

2. LOCALIZATION AND MAPPING

Localization and map building with a monocular camera is very important for autonomous vehicle. For autonomy it should work without any artificial features, but track natural features (such as edges). ORB-SLAM is currently the best visual SLAM method on literature. ORB-SLAM algorithm is developed by Raul Mur-Artal (URL 2) at the University of Zaragoza. The system utilizes ORB descriptors and has several novel features, including

- For all task, ORB-SLAM algorithm uses same ORB features, in this way calculation time decreases.

- When tracking failure, algorithm can recovery itself using previous key frames.
- Automatic map initialization procedure.
- Algorithm can elect the same key frames and features and select best ones for algorithm.

ORB-SLAM is based on ORB descriptors, and it involves three concurrent threads; tracking, mapping and loop-closing (Figure 1). Each of these processes can be executed efficiently with few processor instructions.

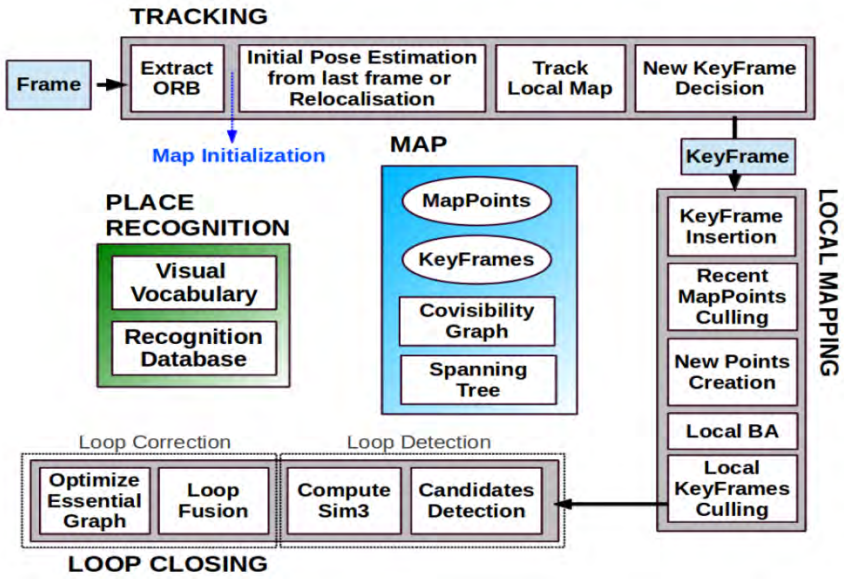

Figure 1: ORB-SLAM algorithm structure (URL 2)

Tracking thread is responsible for camera pose estimation. It does this by searching correspondences of map points in the current and previous frames. It can add a new keyframe in to the graph. First, an initial feature matching with the previous frame is done. Then the pose is optimized using motion-only bundle adjustment techniques. This thread also responsible tracking reinitialization, in case tracking is lost because of any disturbances or other reasons. Reinitialization is done using place recognition module which perform global relocalization.

Local mapping thread runs on every new key frame. Mapping thread builds up the local map by, adding or removing new map points, optimizing key frames on the local map. Upon every new key frame insertion, this thread adds a new node for that key frame and updates all shared points. It the updates the spanning tree, linking this frame with the key frames that has the most points in common. This thread also tries to detect redundant key frames and delete them. Most of the map points have seen at least three key frames or more. If same points are seen least three key frames are deleted.

Loop closing thread takes the last processed key frames and tries to determine whether loop closure has occurred. This is calculated by looking the same ORB feature in map points. If loop closing has been detected, map points errors are calculated and optimization is performed to make correction on points.

ORB-SLAM is also available in the ROS platform; making is a very promising tool in autonomous vehicle and robotics. In this study ORB-SLAM is applied using ROS.

3. SENSOR FUSION

Visual and inertial sensors can work each other very well. Before implementation of sensor fusion algorithm of two visual and inertial sensor, some assumptions are made. Firstly these assumptions are made and then mathematical expression of orientation and position fusion are explained. All the coordinate frames are defined following right hand rule. The earth frame {E} is fixed to world and z_E axis is parallel to gravity vector. IMU sensor frame is defined as {S}. Vision frame (ORB-SLAM) is defined as {V}. World frame of vision frame is {V}. In ORB-SLAM algorithm motion expressed with respect to visual frame {V}. The first assumption is that world frame {E} is very close, even it is same with the visual frame {V}. Other assumption is that projection of x_V axis to $x_E - y_E$ plane is parallel to x_E axis. It is not perpendicular to $x_E - y_E$ plane. With these two assumption,

we can accept two axis looks like same. So we can define filter inertial and visual sensors motion. Figure (2) show coordinate frames.

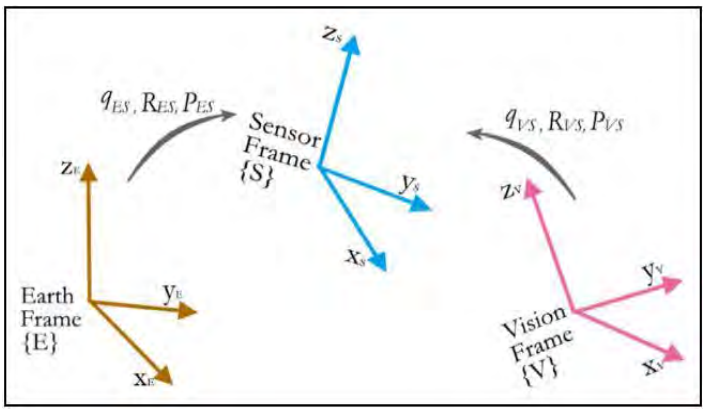

Figure 2: Coordinate frames

Sensor fusion algorithm structure is given figure (3). Algorithm takes rotation q_{ES} from inertial sensor and q_{VS} from visual sensor (ORB-SLAM) algorithm for orientation fusion part. Then apply the linear kalman filter for these measurement. Also Position sensor fusion takes unscaled position $\bar{q}_{VS}$ and rotation q_{VS} measurement from visual sensor, accelerometer measurements a_S from inertial sensor and filtered rotation values q_{ESf} from orientation fusion part. Output of fusion process estimates position and orientation of sensor frame {S} with respect to {E} frame. Sensor fusion is separated two process as seen figure (3). Firstly orientation fusion is estimated, the position fusion is estimated using output of orientation fusion. Linear kalman filter is used for both process to estimate measurements. Separation of kalman filter reduces number of state vector and nonlinearity of system.

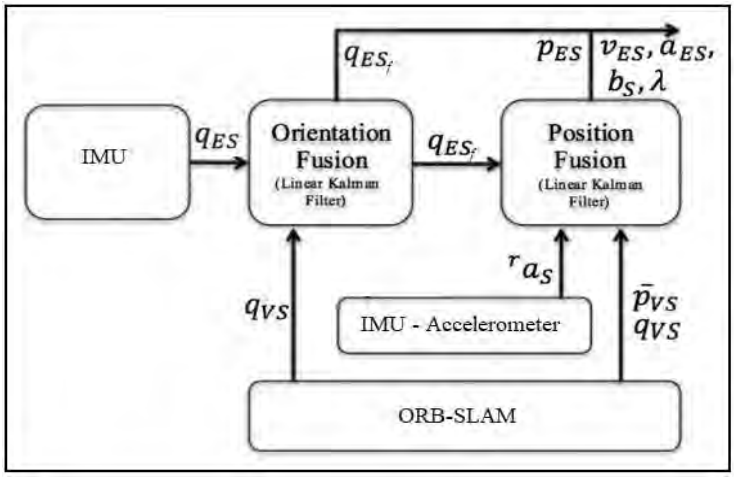

Figure 3: Sensor fusion algorithm structure.

3.1. Orientation Fusion

There is three coordinate systems in the algorithm. Sensor coordinate frame represent the orientation of sensors (visual, inertial). World coordinate frame {E} is reference frame of inertial sensor, visual {V} coordinate frame is reference frame of visual sensor (ORB-SLAM). Purpose of orientation fusion is that obtain the optimum orientation values. In order to obtain optimum orientation values, quaternion measurement of inertial and visual sensors are filtered with kalman filter. Kalman filter is applied with two parts. These are time and measurement update. Sensor reading are not synchronize because of the sensors working speed. Visual sensor (ORB-SLAM) is slower than inertial sensor. So we cannot update sensor measurement at same time. In this situation different

approach is applied to algorithm. When a sensor measurement is available, measurement part is updated with this measurements.

Firstly time update part of kalman filter is applied. Equation (1) and (2) shows time update of kalman filter. Here *A* matrix is state transition matrix and *x* express state vector. Orientation fusion has 4 state parameter equation (3) shows the state vector. These are quaternion values. So we have 4 state for orientation fusion. *A* and *H* matrix are identity matrix. Sensor measurements are directly combined with quaternion values because of sensor measurement also quaternion values.

$$\hat{x}_k^- = A\hat{x}_{k-1} \tag{1}$$

$$P_k^- = AP_{k-1}A^T + Q \tag{2}$$

$$\hat{x} = \begin{bmatrix} q_w & q_x & q_y & q_z \end{bmatrix}^T \tag{3}$$

Here *Q* is process noise covariance matrix and *R* is measurement noise covariance matrix. These covariance matrix should be independent each other. Standard deviations are found for *Q* and *R* matrix;

$$\sigma_Q = 0.4 \tag{4}$$

$$\sigma_R = 0.02 \tag{5}$$

Process and measurement noise covariance matrix are calculated using standard deviation of equation (4) and (5). Both sensors standard deviation are taken same. So both of them has same covariance matrix. Following the time update, measurement update is done with equation (6-8). Here matrix with star '*' express available the sensors measurement. In orientation case, *H* and *R* matrix are same for both sensors measurement.

$$K_k = P_k^- H_*^T (H_* P_k^- H_*^T + R_*)^{-1} \tag{6}$$

$$\hat{x}_k = \hat{x}_k^- + K_k(z_k - H_*\hat{x}_k^-) \tag{7}$$

$$P_k = (I - K_k H_*)P_k^- \tag{8}$$

In order to initialize the orientation fusion procedure, starting values of x_0 and P_0 should be given to algorithm. Here $x_0 = [1\ 0\ 0\ 0]^T$ and P_0 is zero matrix.

3.2. Position Fusion

Position sensor fusion algorithm more complex than orientation fusion. Position fusion algorithm takes 3 input and estimate position. Estimation of orientation q_{ES} with respect to {E} frame, accelerometer measurement a_s with respect to {V} frame and unscaled position $\overline{p}_{VS}$ and orientation q_{vs} measurements are taken from ORB-SLAM algorithm. Position fusion states vector has 13 states; estimation of position p_{ES}, estimation of velocity v_{ES} and estimation of accelerometer a_{ES} values with respect to word frame. Also bias vector and metric scale factor are included inside state vector.

Position fusion algorithm is applied world {E} frame, so all sensor measurement should transferred from {S} frame to {E} frame. In world coordinate system, dynamic acceleration is calculated like equation (9). Here $g_E = \begin{bmatrix} 0 & 0 & 0 & 1 \end{bmatrix}^T$ is gravity vector. Also '$\otimes$' symbol express quaternion multiplication and q_{ES}^* is conjugate of q_{ES}.

$$\begin{bmatrix} 0 \\ a_{ES} \end{bmatrix} = \|a_S\| \left(q_{ES} \otimes a_S \otimes q_{ES}^* - g_E \right) \tag{9}$$

Normalization of accelerometer values;

$$a_S = \frac{a_S}{\|a_S\|} \tag{10}$$

Unscaled position values transferred from {S} sensor coordinate system to world {E} coordinate system;

$$\begin{bmatrix} 0 \\ \overline{p}_{ES} \end{bmatrix} = q_{ES} \otimes \left(q_{VS}^* \otimes \overline{p}_{VS} \otimes q_{VS} \right) \otimes q_{ES}^* \tag{11}$$

After measurement are calculated with respect to world frame. Kalman filter structure is explained for position fusion. Because of asynchronous sensor working, measurement update part of kalman filter is done with available sensor measurement just like orientation fusion. State vector of position fusion algorithm is given equation (12).

$$x = \begin{bmatrix} p_{ES}^T & v_{ES}^T & a_{ES}^T & b_S^T & \lambda \end{bmatrix}^T \tag{12}$$

The state vector is updated every loop, following rule defined by the prediction model, which defines the physics of the inertial system;

$$\dot{p}_{ES} = v_{ES} \tag{13}$$

$$\dot{v}_{ES} = a_{ES} \tag{14}$$

$$\dot{a}_{ES} = n_a \quad \dot{b}_S = n_b \quad \dot{\lambda} = n_\lambda \tag{15}$$

a_{ES} , b_S and λ are modelled as Gaussian normal distribution with zero-mean values;

$$\begin{aligned} p(n_a): \ & N(0, Q_a) \\ p(n_b): \ & N(0, Q_b) \\ p(n_\lambda): \ & N(0, Q_\lambda) \end{aligned} \tag{16}$$

$Q_a = \sigma_a^2 I_3$ and $Q_b = \sigma_b^2 I_3$ are the process noise covariance matrix of filter. After definition of variable we can write the time update of kalman filter. It is same for orientation model time update.

$$\hat{x}_k^- = A\hat{x}_{k-1} \tag{18}$$

$$P_k^- = AP_{k-1}A^T + Q \tag{19}$$

Where A state transition matrix, P is state covariance matrix. Q is process noise covariance matrix and defined as $Q = diag(0_{6x6}, Q_a, Q_b, \sigma_\lambda^2)$.

Measurement update of position fusion process same as orientation fusion. Equation (21-23) can be used to estimate position. We should pay attention to which sensor measurement is reading., since position fusion case measurement model H_* is different for two model and measurement noise v_* also different for each sensor. Here measurement noise is modelled as Gaussian normal distribution;

$$p(v_*): \ N(0, R_*) \tag{20}$$

Measurement model of position fusion;

$$K_k = P_k^- H_*^T (H_* P_k^- H_*^T + R_*)^{-1} \tag{21}$$

$$\hat{x}_k = \hat{x}_k^- + K_k (z_k - H_* \hat{x}_k^-) \tag{22}$$

$$P_k = (I - K_k H_*) P_k^- \tag{23}$$

Measurement model H_* for accelerometer measurement is $H_{as} = (0_{3x6}, I_3, R_{ES}, 0_{3x1})$. Where R_{ES} is rotation matrix and it is calculated with current quaternion angle. Measurement model H_* for vision measurement

is $H_{vs} = (\lambda I_3, 0_{3x9}, p_{ES})$. When acceleration measurement is available, measurement noise covariance matrix will be $R_{as} = \sigma_{as}^2 I_3$. On the other hand, when vision measurement is available, measurement noise covariance matrix will be $R_{vs} = \sigma_{vs}^2 I_3$. Used parameter values are given below table (1). These parameter values are taken from (Liu continued, 2016).

Table 1: Parameter values

Parameter	Value
σ_a	0.5
σ_b	$1 \times e^{-6}$
σ_λ	$1 \times e^{-6}$
σ_{as}	0.013
σ_{vs}	0.005

5. ROS (ROBOT OPERATING SYSTEM)

ORB-SLAM algorithm and sensor fusion process, all work on ROS (Robot Operating System). ROS is not operation system like Windows or Linux, but it is a working environment for robots. It needs to Linux operating system to work. ROS environment is developed from Willow Garage at 2007. It is very effective and dynamic software for robotic application. ROS provide easy and understandable relationships for complex jobs at robotic area. ROS is open source software and provide many package like gazebo simulation tool, rviz visualization tool. In this study rviz package is used for showing vehicle localization performance at experiments.

ROS works with nodes, message, topics and publisher/subscriber. Shortly, there are nodes which responsible different jobs independently. These nodes communicate each other with publishing or subscribing message to specific topics. One node publish message for A topic and another nodes subscribe this A topic and listen message which is sent another nodes. ROS communication structure has many advantages; each nodes can be wrote different programming language and still communicate each other; one of the node is crashed, other node continue to work.

In our case, firstly ORB-SLAM algorithm publish pose of sensor frame to specific topic, same time inertial sensor also publish pose of sensor frame to another topic. Arduino is used to read inertial sensor measurement. Arduino communicate with ROS using rosserial. Related nodes subscribe these topics and sensor fusion algorithm is done. After this process output of sensor fusion algorithm publish to rviz visualization tool. Same time rviz show the algorithm results.

6. EXPERIMENT

Various indoor experiments were performed to test the developed sensor fusion algorithms. Firstly, laptop web cam is used to provide image frames to ORB-SLAM algorithm. An inertial sensor (IMU) is placed top side of the webcam. IMU and Arduino are mounted in the box as seen on figure (4).

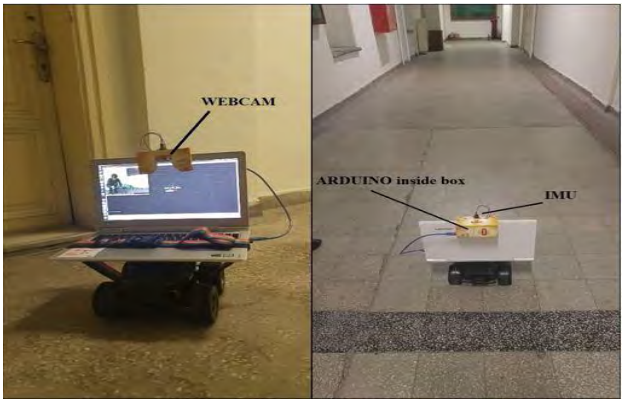

Figure 4: Implementation of system

The experiment is done at indoor environment as seen captured frame at figure (5). Figure (6) shows the screenshot of rviz visualization tool.

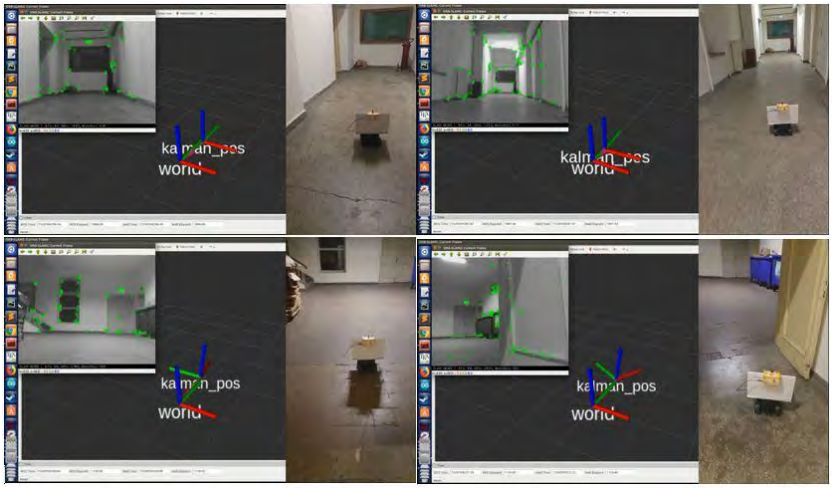

Figure 5: Some captured frame of experiment

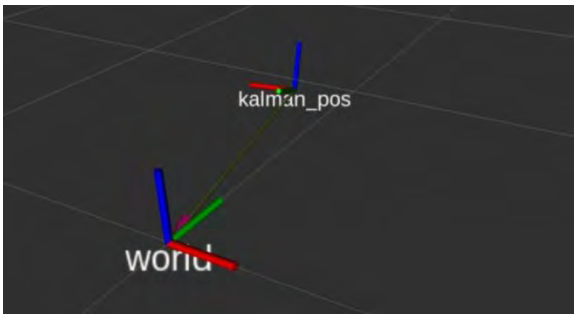

Figure 6: ROS rviz visualization tool screenshot

A RC car is used to carry laptop in this experiment. The path which the car followed looks like 'L'. L shaped path have 20 meter long edge and 7.5 meter short edge. Grid edges distances at rviz is equal to 0.25 meter. Figure 7 shows rviz environment. Each edge of square is 0.25 meter. Total long and short edges of algorithm path approximately are calculated as seen figure (7). Long edge is equal to 0.270 meter and short edge is equal to 0.083 meter. There is a scale factor between real distance and estimated distance as seen results. When scale factor is calculated for long edge, scale factor is obtained as 74.07. Then short edges multiply with scale factor and estimated short edges distance is calculated. It is obtained as 6.14 meter. Our real distance is equal to 7.5 meter. The estimation error is about 1.35 meters

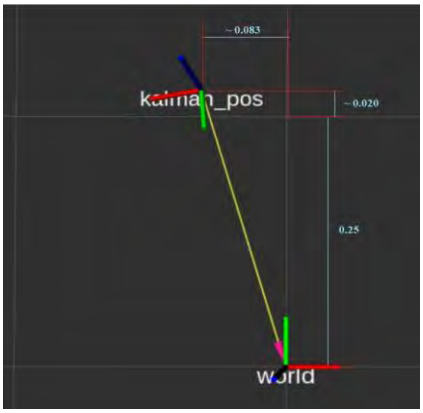

Figure 7: Rviz grid measurement as meter.

.

7. CONCLUSIONS AND FUTURE WORK

Especially in indoor areas, as well as GPS denied environments localization is critical for useful operation of an autonomous vehicle. In this paper, we propose a combination of visual and inertial sensors for autonomous vehicle localization. We have implemented one of the most promising visual odometry algorithms, the ORB-SLAM algorithm, with an inertial measurement unit (IMU) sensor. IMU and ORB-SLAM algorithm are combined for finding optimum localization data. Visual and inertial sensor fusion algorithm has been implemented on a RC car platform for experiments. Visual – inertial sensor model gave us promising results. Algorithm can work real time and follow the path. Estimation result should be multiplied with some scale factor to obtain real estimation. Our algorithm estimates the real position with reasonable error, in other words, the estimation success rate is about 80 percent. One reason of the position error is our approach to model the nonlinear system as linear.

In the future, this algorithm will be integrated to an Odroid XU4 embedded mini-computer for localization.

ACKNOWLEDGMENTS

This research work was supported in part by the Research Projects Office (BAP) of İstanbul Technical University under Grant 38359.

REFERENCES

[1] Bi, Y., Li, J. Qin, H. Lan, M., Shan, M., Lin, F. and Chen, B. M. (2016), "An Mav Localization and Mapping System Based on Dual Realsense Cameras," in International micro air vechicle competition and conference 2016, Beijing, PR of China, pp. 50-55.

[2] Davison, A. J., Reid, I. D., Molton, N. D., and Stasse, O. (2007), "MonoSLAM: Real-Time Single Camera SLAM," in IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 29, No.

[3] Garrido-Jurado, S., Muñoz-Salinas, R., Madrid-Cuevas, F.J., Medina-Carnicer, R. (2016), Generation of Fiducial Marker Dictionaries Using Mixed Integer Linear Programming, Pattern Recognition, Volume 51, March 2016, Pages 481–491.

[4] Klein, D. G., Murray, D., (2007), "Parallel Tracking and Mapping for Small AR Workspaces," in Proc. 6th IEEE and ACM International Symposium on Mixed and Augmented Reality ISMAR, pp. 225-234.

[5] Liu, C., Prior, S.D., Teacy, L. and Warner, M. (2016). Computationally Efficient Visual-Inertial Sensor Fusion for Golabal Positioning System-denied Navigation on Small Quadrotor, Advances in Mechanical Engineering, 8(3), 50 – 55.

[6] Martinez-Carranza, J., Lowens, N., Marquez, F., Garcia, E. O., Mayol-Cuevas, W. (2015), Towards Autonomous Flight of Micro Aerial Vehicles using ORB-SLAM, Workshop on Research, Education and Development of Unmanned Aerial Systems (RED-UAS), November 23-25, 2015, Cancun, Mexico, pp. 241-248.

[7] Muñoz-Salinas, Rafael, Manuel J. Marín-Jiménez, Enrique Yeguas-Bolivar and Rafael Medina Carnicer. (2016), "Mapping and Localization from Planar Markers." *CoRR*abs/1606.00151.

[8] Mur-Artal, R., Montiel, J. M. M. and Tardós, J. D. (2015), ORB-SLAM: A Versatile and Accurate Monocular SLAM System. IEEE Transactions on Robotics, vol. 31, no. 5, pp. 1147-1163

[9] URL 1: MTi User Manual, Xsens, 27 February. http://www.farnell.com/datasheets/1935846.pdf, 06,04,2018.

[10] URL 2: Real-Time SLAM for Monocular, Stereo and RGB-D Cameras, with Loop Detection and Relocalization Capabilities, available at https://github.com/raulmur/ORB_SLAM2

[11] Yiğit, Bora, Altuğ, Erdinç. (2012), "Visual Attitude Stabilization of a Unmanned Helicopter in Unknown Environments with an Embedded Single-board Computer", IEEE International Symposium on Robotic and Sensors Environments (ROSE), pp. 49 – 54.