Autonomous Ground Vehicle Navigation Using a Novel Visual Positioning System

Sergio Reyes
UMI-LAFMIA
CINVESTAV
Mexico City, Mexico
sergio.reyes.sanchez1@gmail.com

Oscar F. Tokunaga
UMI-LAFMIA
CINVESTAV
Mexico City, Mexico
oscar.tokunaga@cinvestav.mx

Eduardo S. Espinoza
UMI-LAFMIA CINVESTAV
CONACYT
Mexico City, Mexico
eduardo.espinoza@cinvestav.mx

Sergio Salazar
UMI-LAFMIA
CINVESTAV
Mexico City, Mexico
sergio.salazar.cruz@gmail.com

Hugo Romero
CITIS
UAEH
Hidalgo, Mexico
rhugo@uaeh.edu.mx

Rogelio Lozano
UMI-LAFMIA
CINVESTAV
Mexico City, Mexico
rogelio.lozano@hds.utc.fr

***Abstract*—In this paper a novel visual positioning system for slip correction of vehicles in GPS-denied environments based on Kalman filter is presented. Kalman filter fuses 2 sources of data; the first source of data is obtained from odometry, while a computer vision algorithm is used to obtain the second source of data. Using a plane-based pose estimation algorithm that employs four reference points and the Consensus-based Tracking and Matching algorithm (CMT) we were able to obtain pose estimation with centimeter accuracy. The four reference points correspond to the pixel coordinates of the corners of a rectangle closing a region in the roof where the camera should be pointing. Since the employed camera is mounted into the vehicle and it has eye fish lens, an image distortion correction method is used. As the CMT algorithm is robust to scale and rotations, position and orientation (POSE) estimation using computer vision is obtained even if the roof is not completely visible. Autonomous navigation of a ground wheeled vehicle was achieved using the proposed algorithm performing different courses on long period of time and no big slip was observed.**

***Index Terms*—Data Fusion, Odometry, Visual Odometry, CMT, Autonomous Navigation.**

I. INTRODUCTION

An autonomous vehicle is a robot capable of navigating without human input and sensing its environment by means of an autopilot system assisted by different advanced computational tools and sensors such as computer vision, odometry, infrared sensors, radar and global positioning systems (GPS). The autopilot system relies on advanced control systems that interpret sensory information to identify appropriate navigation paths, as well as obstacles and relevant signals. Nowadays, we can find these kind of systems in ground, sea or air vehicles, being the latter ones, recently used in micro aerial vehicles (MAVs) or drones.

The use of computer vision for the development of different solutions especially on autonomous navigation has had a great growing, from visual assistance for parking until collision avoidance systems. Numerous algorithms have been developed to perform tasks that contribute to the development of more intelligent autonomous vehicles. Some of the algorithms employed to perform tasks as lane detection [1], [2] or visual odometry [3], [4] have helped to enhance the autopilot systems in order to achieve better navigation results. Nevertheless, new algorithms require extensive data processing or, in most of the cases, they only work under certain conditions or in controlled environments.

Despite recent advances in autonomous car technologies, tasks as keeping the vehicle in the correct lane by using computer vision is still a challenging task, since, in the case of outdoors environments, quality of the available images can be totally affected by external factors, for instances: bad conditions like pointing the light directly to the camera lens, side wind effects, sun reflections or high velocities affect the desired behavior, reason why the information of precise vehicle position is required to achieve satisfactory results regardless of weather conditions.

In the literature there are many map-based vehicle localization algorithms developed to try to keep a vehicle along the desired trajectory. This algorithms are frequently employed due to the availability of i) accurate digital navigation maps, which are combined with the information obtained from onboard cameras used to detect the road markings by performing estimation of the lateral and directional deviations using Kalman filters [5], or ii) sensors such as GPS or zenith cameras, simultaneous localization and mapping (SLAM) [6], used to obtain global positions in the environment. However, a large variety of applications for autonomous vehicles would rely on the autonomous capabilities in GPS denied environments to handle difficult, dirty or dangerous jobs. Similarly, a fully autonomous operation of autonomous ground vehicles requires real-time obstacle detection as well as collision-avoidance trajectory planning. Early autonomous vehicles employed 2D laser scanners in addition of vision systems for autonomous

This work has been partially supported by the Mexican Conacyt Project LN295536.

navigation, with a camera as the primary sensor [7], [8], whereas most recent detection systems use different sensors to achieve more complex tasks such as omni-directional perception of obstacles [9].

In this work autonomous navigation of a ground vehicle was achieved by using a data fusion algorithm which guarantees lane following by employing the trajectory created from $X-Y$ points without the necessity of lane detection.

The main contribution of the proposed algorithm is related to the ability of the CMT algorithm to estimate the POSE and the employ of the Kalman filter for data fusion to estimate the local position of an autonomous vehicle while achieving automatic drift correction caused by integration methods and environmental effects.

The remaining of the paper is organized as follows: Section II presents the methodology for obtaining the necessary sources of information for data fusion. Section III presents the proposed algorithm to estimate the POSE using computer vision and data fusion, while in Section IV we analysed the effectiveness of the proposed algorithm. Finally, Section V summarizes the conclusions and future directions of the work presented in this paper.

II. BACKGROUND

A. Odometry

Vehicles usually have an odometer or there is a way to obtain odometry information, mostly by using sensors in the wheels, which provide the traveled distance based on the wheel diameter and in combination with a compass or an Inertial Measurement Unit (IMU). This distance can be split into its X and Y components, obtaining an estimated relative position as well as an estimation of the velocity, however, these estimations could cause the vehicle slips or drifts along the path. For this reason, slips need to be compensated, so, many techniques have been developed like the reported in [10], [11], where authors used GPS information in order to try to overcome this issue.

1) Visual Odometry and Localization: This localization strategy involves the use of computer vision algorithms and various strategies with different results have been developed. One alternative is by using optical flow, which allows to estimate the velocity of an object's movement. Since there are two sources of movement that generates optical flow components (rotational and translational), each source of movement needs to be identified to compensate the drift on the estimated position. Normally, the flow components are obtained by using feature extraction and this generally requires lot of computing calculations. Similarly, another factor to be compensated is the altitud and some solutions are presented in [12], [13].

Another approach for locating the robot in a 3D space is the pose estimation using stereo vision and SLAM, but this technique is very expensive in computational terms, so it requires a big processor with a big graphic card, and many times the robots are not capable to carry that type of hardware (more roughly speaking: MAVs). Recently, some light solutions have been evaluated in embedded computers like the works presented in [14] and [15], but them require the implementation of more sophisticated data fusion algorithms.

B. Consensus-based Tracking and Matching algorithm (CMT)

CMT is an algorithm designed for features matching and objects tracking. This algorithm allows to track an object even if it has a deformation or if it has partial occlusions. The center of the region is calculated in order to estimate the position of the object in the image. Furthermore, this algorithm is robust against scale or rotations, and it does not require previous training since it only needs the current image key-points.

The algorithm bases the object model on a set of key-points where each characteristic point denotes a location $r \in \mathbb{R}^2$ and a descriptor f. Binary descriptors $f \in [0,1]^d$ are used in order to have a better computational performance and be able to execute the code in an embedded system, an object O is initialized with the detection and description of the characteristic points in an image I_1 within the b_1 region, followed by the normalization-average of the location of each characteristic point, later the Hamming distance between them is calculated

$$d(f^1, f^2) = \sum XOR(f_i^1, f_i^2) \tag{1}$$

In order to do not lose the characteristic points and to carry out an optimal follow-up, the displacement of each one of the points in K_{t-1} from I_{t-1} to it, is calculated using a pyramidal variant of the Lukas-Kanade optical flow method. Then, a voting of each characteristic point is made based on the distance of each point towards the center of the object, involving the scaling and rotation of the object [16].

C. Rigid Transformation

In order to find the rotation and translation between two sets of 3D points, a rigid transformation needs to be applied. Rigid transformation (2) includes rotations, translations and reflections. In kinematics a rigid transformation in Euclidean 3D space is used to represent the angular and linear displacement of rigid bodies (with constant shape and size).

$$B = RA + t \tag{2}$$

Fig. 1 shows the rotation R and translation t which are used to align the object position A relative to B with the minimum error in terms of least squares.

To find the rigid transformation, the object centroid (3) is required. Since the centroid is the geometrical center, calculating the average value of the set of vertices of the object solves the problem, and it can be obtained as:

$$C_{A,B} = \frac{1}{N}\sum_{i=1}^{N} P_{A,B}^i \tag{3}$$

where $C_{A,B}$ are the centroids of the vertices $P = [x\ y\ z]$ that belongs to A and B, respectively, then using a singular value decomposition SVD with the accumulated matrix H, we can find the optimal rotation (6):

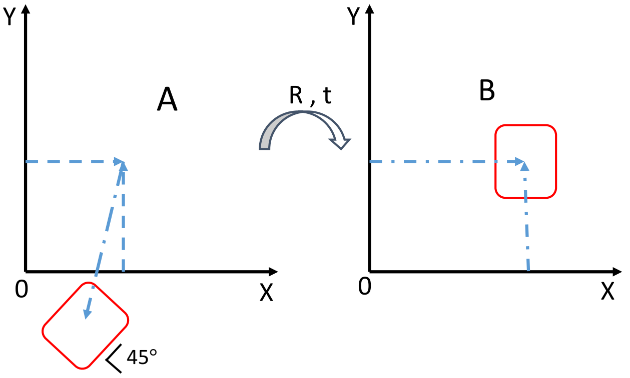

Fig. 1. Rigid transformation applied to the object.

$$H = \sum_{i=1}^{N} (P_A^i - C_A)(P_B^i - C_B)^T \quad (4)$$

$$[U, S, V] = SVD(H) \quad (5)$$

$$R = VU^T \quad (6)$$

where R is the rotation matrix $m \times p$, U is a real or complex unitary matrix $m \times m$, S is a rectangular-diagonal matrix $m \times p$, and V is a unitary matrix $p \times p$ of which S denotes an escalation matrix and U, V are rotation matrices

Finally the translation between the two sets of 3D points is given by the difference between the centroid of the rotated object and the centroid of the displaced object as [17]

$$t = -RC_A + C_B \quad (7)$$

III. LOCALIZATION AND DATA FUSION ALGORITHMS

The proposed algorithm was implemented in an experimental setup composed by a ground wheeled vehicle and a set of drawings fixed on the roof of the experimental room as a source of localization. In this work we perform data fusion between the POSE obtained from a video camera and the odometry obtained from the vehicle's wheels.

A. Localization Algorithm

On the first hand, odometry can be used to obtain the distance traveled by the vehicle by implementing an integration method and combining the heading angle ψ of the vehicle with the velocity V_t. We used equation (8) and equation (9) in order to obtain the traveled distances p_x and p_y in the coordinate system shown in Fig. 2.

$$\begin{aligned} p_x(k) &= p_x(k-1) + \quad V_t \cos(\psi) \quad (8) \\ p_y(k) &= p_y(k-1) + \quad V_t \sin(\psi) \quad (9) \end{aligned}$$

It is well known that integration methods are not accurate because of the drift on the estimation along the time, so that drift needs to be compensated for performing precision navigation.

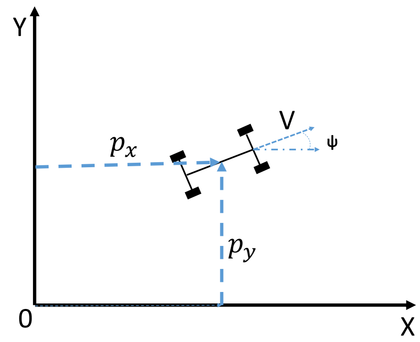

Fig. 2. Coordinate system used for the derivation of traveled vehicle's distances.

On the other hand, we use the CMT algorithm [16] and the method presented in [18] to obtain the POSE estimations by using a fish eye camera pointing to the roof of the experimental room. In [18] authors describe that a full POSE (3 axis position and 3 axis orientation) can be estimated using only four points which distances between them are known. Therefore, a set of points was selected within a rectangular region previously measured on the roof of the experimental room, so we were able to know the distance between the four points required to solve the POSE problem.

Considering only the $X - Y$ plane, equation (10) defines the projection of the image points to the real points. In this form, the matrix that contains the rotational part R_i, and the translational part t_i of the projection, needs to be solved using the known parameters which correspond to the real distance between the four corners of the selected region. The pixel positions corresponding to the four corners are used by the SVD decomposition to solve the matrix that contains the POSE of the camera with respect to the roof of the experimental room [16].

$$Q_{jk}^w = \begin{pmatrix} R_i & t_i \\ 0^T & 1 \end{pmatrix} \begin{pmatrix} X_{jk} \\ Y_{jk} \\ 0 \\ 1 \end{pmatrix} \quad (10)$$

In this work we used a fisheye camera for image acquisition and the algorithm presented in [19] and [20] was used to correct the distortion of the big angular lens view. In this way, the camera calibration matrix K required in [18] can be simplified with the identity matrix.

B. Data Fusion Algorithm

We used the Kalman filter for data fusion since it provides better results compared with other approaches like the complementary filter. In this sense, we considered two cases: i) when the data obtained from the camera is null or deficient and, ii) when the data obtained from the camera is accurate or available. In the first case the best velocity estimation is the one obtained with the odometer, while the estimated position is given by

$$P_k = P_{k-1} + V_k \Delta t \tag{11}$$

where P_k is the position vector that contains p_x and p_y, P_{k-1} is the accumulative value of the position on the previous step, V_k is the velocity vector that contains velocities v_x and v_y, respectively, and Δt is the integration step size.

For the second case, when the POSE obtained from the camera image is available and accurate, the model can bee seen as an observer since the estimated position can be seen as an error between positions. In this case, the estimated velocity is given as

$$P_k = P_{k-1} + V_k \Delta t - V_{est} \Delta t \tag{12}$$

Finally, when the image is available, then the Kalman filter is updated.

The Kalman filter is used to perform the data fusion between the odometry an the POSE obtained from the plane-based algorithm. The implementation of the Kalman filter was divided into two steps: i) prediction and, ii) measurements update.

The mathematical model used is given by

$$x_k = A x_{k-1} + B u_k + w_k \tag{13}$$

where $x_{k-1} \in \mathcal{R}^{n\times 1}$ is the previous state of the system, $A \in \mathcal{R}^{n\times n}$ is the state transition matrix defined in (15), $B\in \mathcal{R}^{n\times m}$ is the input matrix given in (16), $u_k\in \mathcal{R}^{m\times 1}$ is the control input, and $x_k\in \mathcal{R}^{n\times 1}$ is the state of the system which is defined as follows

$$x = \begin{bmatrix} x \\ y \\ \dot{x}_d \\ \dot{y}_d \end{bmatrix} \tag{14}$$

where x and y are the positions of the car in the $X-Y$ plane, respectively, and $\dot{x}_d$ and $\dot{y}_d$ are the velocities with the drift induced by the integration method, which causes the deviation on the position along the time.

$$A = \begin{bmatrix} 1 & 0 & -\Delta t & 0 \\ 0 & 1 & 0 & -\Delta t \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{15}$$

$$B = \begin{bmatrix} \Delta t & 0 \\ 0 & \Delta t \\ 0 & 0 \\ 0 & 0 \end{bmatrix} \tag{16}$$

$u = [\dot{x}\ \dot{y}]$ and $w_k\in \mathcal{R}^{n\times 1}$ is the Gaussian noise with covariance $Q_k\in \mathcal{R}^{n\times n}$ given as

$$Q_k = \begin{bmatrix} 0.01 & 0 & 0 & 0 \\ 0 & 0.01 & 0 & 0 \\ 0 & 0 & Q_1 & Q_2 \\ 0 & 0 & Q_3 & Q_4 \end{bmatrix} \Delta t \tag{17}$$

The model used in the step of updating the measurements is given by:

$$z_k = H x_k + v_k \tag{18}$$

where $z_k\in \mathcal{R}^{n\times 1}$ is the observed state, $v_k\in \mathcal{R}^{n\times 1}$ is the measurements noise, $H\in \mathcal{R}^{m\times n}$ is the measurement model containing only position measurements that come from the camera pose estimation:

$$H = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \end{bmatrix} \tag{19}$$

v_k is the measurement noise with covariance $R\in \mathcal{R}^{m\times m}$

$$R = \begin{bmatrix} r_{00} & r_{01} \\ r_{10} & r_{11} \end{bmatrix} \tag{20}$$

1) Prediction: The equations used in this step are given as

$$\hat{x}_k^- = A\hat{x}_{k-1} + Bu \tag{21}$$

where $\hat{x}_k^-\in \mathcal{R}^{n\times 1}$ is a-prior state estimation and an a-prior error covariance matrix is needed $P_k^-\in \mathcal{R}^{n\times n}$, which is given as

$$P_k^- = A P_{k-1} A^T + Q_k \tag{22}$$

2) Measurements Update: Performing a comparison between the measurements and the computed state using the model H helps to update the estimated state

$$\tilde{y}_k = z_k - H\hat{x}_k \tag{23}$$

where the estimation error $\tilde{y}_k\in \mathcal{R}^{n\times 1}$. Then the next step is to update the covariance using the next equation

$$S_k = H P_k H^T + R \tag{24}$$

To update the error covariance we used

$$P_k = (I - K_k H + R) P_k^- \tag{25}$$

where $I\in \mathcal{R}^{n\times n}$ is the identity matrix and $P_k\in \mathcal{R}^{n\times n}$.

Finally, the Kalman filter gain is obtained with

$$K_k = P_k H^T S_k^{-1} \tag{26}$$

The Kalman gain indicates which model is more accurate, either the estimated model or the measured model, and then, we calculate the final estimated state as

$$\hat{x}_k = \hat{x}_k^- + K_k \tilde{y}_k \tag{27}$$

This algorithm is based on the methodology presented in [21].

IV. IMPLEMENTATION AND EXPERIMENTAL RESULTS

The proposed algorithm was tested in an autonomous wheeled vehicle developed by the University of Berlin. The vehicle has the following sensors and processors: gyroscopes, accelerometers, motor encoders, a stereo camera pointing to the front of the vehicle, one fish-eye camera pointing to the roof, an Odroid XU-4 companion computer and an Arduino nano.

The platform runs Ubuntu Linux 14.04 with ROS Indigo and OpenCV 2.4. We used both Python as well as C++ for programming the proposed algorithm in the ROS platform.

Algorithm 1 summarizes all the necessary steps required to implement the proposed methodology previously described in Section III.

Algorithm 1 Data fusion algorithm and vehicle behavior

Require: Odroid with ROS and Ocam functions installed, eye-fish camera, odometer and IMU connected

```
Initialize the system, IMU and CMT ROI with preloaded image
Using a command sent via ROS UsbJoy we select the action to perform (to follow a loaded trajectory or starting to record a new trajectory)
if Follow a trajectory then
    if CMT has enough ROI key-points then
        Calculate POSE using the 4 points and pass the data via ROS message to data Fusion node
    end if
    if POSE data is available then
        The data fusion node using the odometry and yaw angle determines the x-y velocities components by performing the calculations using the kalman filter
    else
        Only performs an integration of odometer velocity
    end if
    The vehicle follows the trajectory
else
    wait for command or reacts to manual control commands
end if
```

Fig. 3 depicts the set of drawings that were fixed on the roof of the experimental room in order to have enough key-points for the implementation of the proposed algorithm. Despite a minimum number of points is not required, having more key-points on the region of interest means more accuracy, even if not all the points are detected. We defined a trigger on the software based on the percentage of active key-points (we found that 30% or more is the best value) indicates if the POSE estimation is available or accurate enough and then, a drift correction for the odometry is applied.

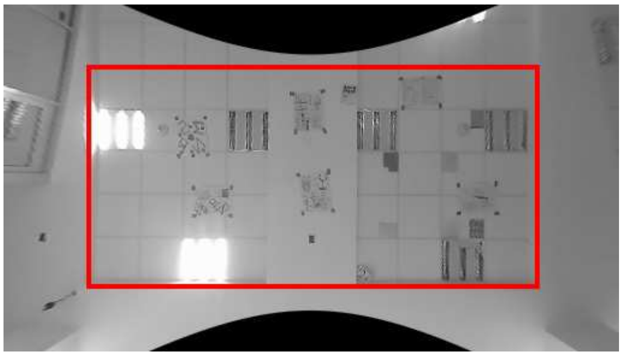

Fig. 3. Region enclosing the key-points: the distance between the four corners is known and it is used to solve the proposed algorithm.

The four points used for plane-based POSE estimation are the pixel coordinates of the rectangle's corners enclosing the drawings shown in Fig. 3. Since the CMT algorithm is robust against occlusions, the pixel position is obtained even when only one part of the region of interest is seen by the camera. The process from the image is captured and it is processed on the Odroid computer takes less than 100ms, so if the number of key-points is enough to obtain a good correction, the Kalman filter is updated.

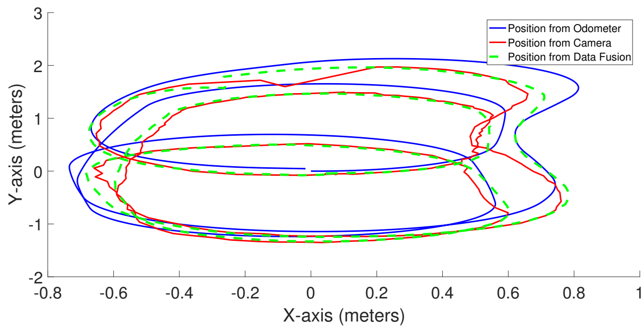

Fig. 4. Comparison between the positions obtained from the camera (red), odometry (blue) and the fusion of both positions (green).

The resulting position obtained from data fusion is observed on Fig. 4, this position is almost on the middle of camera and odometry resulted position. The filtered data was used to perform a trajectory tracking controller, while the desired path was previously driven and the positions were recorded as a set of points. Once the desired path has been defined, it will be used as the reference to make path following in autonomous mode. Fig. 5 shows the car that was used to perform the experimental tests which include lane following using the proposed algorithm obtaining centimeter-level accuracy.

A video of the trajectory following as well as the lane following experiments can found on

Fig. 5. Autonomous ground vehicle used for lane following tests without seeing the lane.

https://youtu.be/9y9LkQayVTY

V. CONCLUSIONS AND FUTURE WORK

Autonomous navigation by using the proposed algorithm was achieved. This algorithm could be applied on GPS denied environments or environments where infrastructure does not change constantly. The achieved accuracy is related to the camera features. Since the CMT algorithm is robust against scale and rotations, a camera stabilizer was not required. Finally, since the proposed algorithm allows to estimate the 3 axis position and orientation it can be applied to more complex systems such as aerial vehicles.

Lane detection can be used as a third source of data to enhance the task of keeping the vehicle in the lane. In future tests we will try to include on the filter the lane information obtained from a frontal camera.

The vehicle velocity was limited to 1.5 m/s to allow processing data and to avoid big trajectory deviation, on the performed experiments $\pm 5cm$ precision was reached. To increase the vehicle velocity and enhance the localization algorithm results, it is needed to perform the CMT algorithm and to estimate the POSE faster, this can be done using the Odroid XU-4 GPU feature, which is enabled to use OpenCL allowing to execute graphic calculation faster, this feature will be tested as future work.

ACKNOWLEDGMENT

This work was partially supported by the Mexican National Council for Science and Technology Project *Laboratorio Nacional en Vehículos Autónomos y Exoesqueletos 295536*.

REFERENCES

[1] C. Y. Low, H. Zamzuri and S. A. Mazlan, "Simple robust road lane detection algorithm," 2014 5th International Conference on Intelligent and Advanced Systems (ICIAS), Kuala Lumpur, 2014, pp. 1-4. doi: 10.1109/ICIAS.2014.6869550

[2] S. Hilsenbeck, A. Mller, R. Huitl, G. Schroth, M. Kranz and E. Steinbach, "Scale-preserving long-term visual odometry for indoor navigation," 2012 International Conference on Indoor Positioning and Indoor Navigation (IPIN), Sydney, NSW, 2012, pp. 1-10.

[3] T. Mouats, N. Aouf, L. Chermak and M. A. Richardson, "Thermal Stereo Odometry for UAVs," in IEEE Sensors Journal, vol. 15, no. 11, pp. 6335-6347, Nov. 2015.

[4] C. Fu, A. Carrio and P. Campoy, "Efficient visual odometry and mapping for Unmanned Aerial Vehicle using ARM-based stereo vision pre-processing system," 2015 International Conference on Unmanned Aircraft Systems (ICUAS), Denver, CO, 2015, pp. 957-962.

[5] D. Gruyer, R. Belaroussi and M. Revilloud, "Map-aided localization with lateral perception," 2014 IEEE Intelligent Vehicles Symposium Proceedings, Dearborn, MI, 2014, pp. 674-680.

[6] H. Durrant-Whyte and T. Bailey, "Simultaneous localization and mapping: part I," in IEEE Robotics and Automation Magazine, vol. 13, no. 2, pp. 99-110, June 2006.

[7] A. Bachrach, A. de Winter, R. He, G. Hemann, S. Prentice and N. Roy, "RANGE - robust autonomous navigation in GPS-denied environments," 2010 IEEE International Conference on Robotics and Automation, Anchorage, AK, 2010, pp. 1096-1097.

[8] D. Scaramuzza et al., "Vision-Controlled Micro Flying Robots: From System Design to Autonomous Navigation and Mapping in GPS-Denied Environments," in IEEE Robotics and Automation Magazine, vol. 21, no. 3, pp. 26-40, Sept. 2014.

[9] Droeschel, David and Nieuwenhuisen, Matthias and Beul, Marius and Holz, Dirk and Stckler, Jrg and Behnke, Sven, "Multilayered Mapping and Navigation for Autonomous Micro Aerial Vehicles", Journal of Field Robotics, volume 33, doi:10.1002/rob.21603, pp. 451-475, 2016

[10] K. Ohno, T. Tsubouchi and S. Yuta, "Outdoor map building based on odometry and RTK-GPS positioning fusion," Robotics and Automation, 2004. Proceedings. ICRA '04. 2004 IEEE International Conference on, 2004, pp. 684-690 Vol.1.

[11] R. Toledo-Moreo, M. A. Zamora-izquierdo and A. F. Gomez-skarmeta, "IMM-EKF based Road Vehicle Navigation with Low Cost GPS/INS," 2006 IEEE International Conference on Multisensor Fusion and Integration for Intelligent Systems, Heidelberg, 2006, pp. 433-438.

[12] R. Ross, J. Devlin and S. Wang, "Toward Refocused Optical Mouse Sensors for Outdoor Optical Flow Odometry," in IEEE Sensors Journal, vol. 12, no. 6, pp. 1925-1932, June 2012.

[13] W. Chen, H. Fu, M. Shi and Y. Chen, "A Dense Optical Flow-Based Feature Matching Approach in Visual Odometry," 2017 9th International Conference on Intelligent Human-Machine Systems and Cybernetics (IHMSC), Hangzhou, 2017, pp. 343-348

[14] Z. Wang, Q. Jia, P. Ye and H. Sun, "A depth camera based lightweight visual SLAM algorithm," 2017 4th International Conference on Systems and Informatics (ICSAI), Hangzhou, China, 2017, pp. 143-148.

[15] C. Fu, A. Carrio and P. Campoy, "Efficient visual odometry and mapping for Unmanned Aerial Vehicle using ARM-based stereo vision pre-processing system," 2015 International Conference on Unmanned Aircraft Systems (ICUAS), Denver, CO, 2015, pp. 957-962.

[16] N. Georg, P. Roman, "Clustering of Static-Adaptive Correspondences for Deformable Object Tracking" 2015 IEEE Computer Vision and Pattern Recognition (CVPR)

[17] P. J. Besl and N. D. McKay, "A method for registration of 3-D shapes," in IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 14, no. 2, pp. 239-256, Feb. 1992.

[18] P. Sturm, "Algorithms for plane-based pose estimation," Proceedings IEEE Conference on Computer Vision and Pattern Recognition. CVPR 2000 (Cat. No.PR00662), Hilton Head Island, SC, 2000, pp. 706-711 vol.1.

[19] Scaramuzza, D., Martinelli, A. and Siegwart, R.. "A Flexible Technique for Accurate Omnidirectional Camera Calibration and Structure from Motion", Proceedings of IEEE International Conference of Vision Systems (ICVS'06), New York, January 5-7, 2006.

[20] Scaramuzza, D., Martinelli, A. and Siegwart, R.. "A Toolbox for Easy Calibrating Omnidirectional Cameras", Proceedings to IEEE International Conference on Intelligent Robots and Systems (IROS 2006), Beijing China, October 7-15, 2006.

[21] Welch, Greg & Bishop, Gary. (2006). An Introduction to the Kalman Filter. Proc. Siggraph Course. 8.