A Fast Algorithm for Robot Localization Using Multiple Sensing Units

Reinier Oves García(✉), Luis Valentin, José Martínez-Carranza, and L. Enrique Sucar

Computer Science Department,
Instituto Nacional de Astrofísica Óptica y Electrónica,
Sta. María Tonantzintla, 72840 Puebla, Mexico
{ovesreinier,luismvc,carranza,esucar}@inaoep.mx

Abstract. This paper presents a fast algorithm for camera selection in a robotic multi-camera localization system. The scenario we study is that where a robot is navigating in an indoor environment using a four-camera vision system to localize itself inside the world. In this context, when something occludes the current camera used for localization, the system has to switch to one of the other three available cameras to remain localized. In this context, the question that arises is that of "what camera should be selected?". We address this by proposing an approach that aims at selecting the next best view to carry on the localization. For that, the number of static features at each direction is estimated using the optical flow. In order to validate our approach, experiments in a real scenario with a mobile robot system are presented.

Keywords: Multi-camera navigation · Multi-camera localization Guidance

1 Introduction

Nowadays, by using conventional cameras it is possible to obtain a set of images, which can be processed in order to obtain the estimated position of the robot in real time [7]. However, multi-camera approaches have also been proposed as having more than one camera observing different parts of the scene and constitute an attractive approach that can be helpful when autonomous navigation is performed. One application of these multi-camera approaches can be found in [15], where more than two cameras are used for eliminating motion ambiguity problems in a visual odometry system.

Motivated by the advantages of using multi-camera systems capable of capturing an approximate 360° field of view, in this work we explore the scenario of when the localization system is partially or totally occluded in the current active view, this is, in one of the cameras that is currently being used for feature tracking and localization. In this scenario, one of the available cameras has to be selected to avoid tracking loss. Motivated by this, we propose an efficient method for camera selection aiming at maintaining localization in the event of camera occlusion.

Our approach is based under the assumption that only one view is used for localization while the rest of the cameras are used for relocalization in case an obstruction in the main view is presented. Therefore, the contribution of this work is two fold: (i) a methodology based on the optical flow exhibited by the scene structure w.r.t. is presented in order to estimate the velocity of the visual texture observed by the camera and use it to distinguish motion from steadiness; and (ii) we present a histogram-based approach in order to quantify the evolution of the texture's motion frame by frame. This evolution is assessed in terms of how steady or unsteady scenes are along the time.

In order to present our contribution, this paper is organized as follows: Sect. 2 presents the related work; Sect. 3 provides a description of our system. In Sect. 4 the proposed algorithm is described while in Sect. 5 experiments are conducted in order to make clear the idea of this paper. Finally, in Sect. 6 conclusion and future work are included.

2 Related Work

Arguably, a multi-camera rig sensor may arises as a better choice than using an omni-directional camera [18] to address different problems, i.e. the localization [14]. The latter is due to the fact that several conventional cameras mounted in a rig can be set up to obtain a wider field of view. In contrast, an omni-directional system may have a superior field of view but at the expense of exhibiting a strong distortion, where calibration and measurement process are not straightforward.

In [15] the authors use a multi-camera stereo rig to solve motion ambiguity problems in their visual odometry process. In [12], a framework is described for 6D absolute scale motion and structure estimation for a stereo multi-camera system with non-overlapping fields of view in indoor environments. In [2] the authors introduces a testbed for sensor and robot network systems composed of 10 cameras and 5 mobile robots for self-localization and obstacle avoidance using machine vision and wireless communication. In [10] the authors present an extension of the monocular ORB-SLAM for multiple cameras alongside an inertial measurement unit *(IMU)* and a multi-camera SLAM is proposed in [11] based on a probabilistic approach for data association, that takes into account that features can also move between cameras under robot motion.

Several visual SLAM algorithms use keyframes to reduce the computational cost for developing online optimization. Entropy handling in the keyframes insertion improves significantly the system's ability to localize. This approach is recently presented by [5] and is implemented within the omni-directional multi-camera parallel tracking and mapping framework. Another interesting recent work is proposed by Harmat *et al.* [9]. They addressed the pose problem for UAV's using Multiple fish-eye cameras for tracking and mapping a small UAV in unstructured environment systems. Their approach improves the PTAM [13] pose estimation with the multi-camera rig.

Besides of localization, a multi-camera stereo rig may be used to address another kind of problems, like in the work proposed by Akash *et al.* [1], where a method for performing a fast 6-DOF head pose tracking using a cluster of rolling shutter cameras is proposed in order to deal with end-to-end latency challenge in Augmented Reality/Virtual Reality *(AR/VR)* applications.

3 System Overview

The architecture proposed in this paper counts of two parts (i) the *Guidance* sensor (see Fig. 1(a)) and (ii) a service robot (see Fig. 1(b)). Technically, the *Guidance* sensor is an upgraded version of Zhou et al.'s work [19] which is a visual mapping solution based on four cameras and a single processing chip-Altera's SoC FPGA. In our case, the *Guidance* is a multi-camera rig that captures up to 5 stereo pairs with a depth image associated to each stereo pair at a frequency of 18 fps.

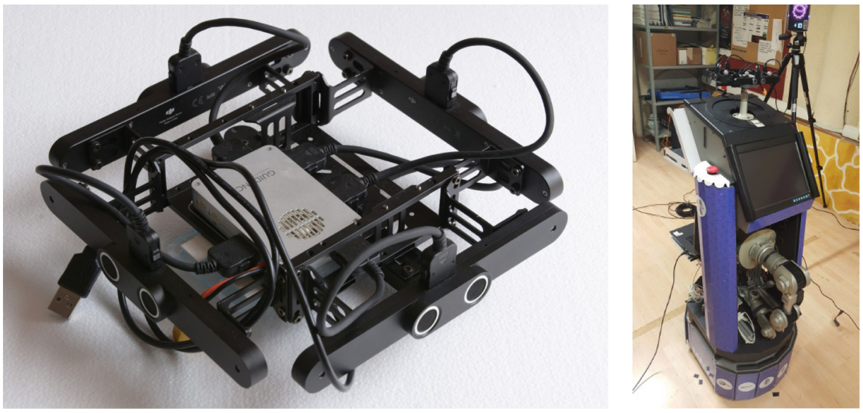

(a) Guidance sensor. (b) Robot Sabina.

Fig. 1. (a) *Guidance* sensor: this image shows four of its five *stereo + ultrasound* cameras, which return gray and depth images, the sensor can be used to observe the scene in almost 360°. (b) Robot Sabina with the *Guidance* sensor on the top of it.

The way camera units are located in the rig enables the observation of the world in five directions *(front, back, left, right and top)*. The SDK, made available by the manufacturer [8], enables acquiering up to 10 gray images *(from the 5 stereo pairs)* simultaneously, with the caveat that only 2 depth images can be accessed simultaneously at a frequency of 18 Hz. Considering that we are interested in multi-camera localization within dynamic environments where laser-odometry may not be sufficient, we test our algorithm using a service robot based on a PatrolBot platform. The platform has a sonar ring, two wheels with independent motors with encoders, a Laser SICK LMS200, a video camera Canon

VCC5, speakers, and an integrated PC. The integration of this novel visual sensing platform with our multi-threading probabilistic visual odometry framework allows us to estimate the robot's localization in a more accurate way.

4 Velocity Map for Camera Selection

The algorithm proposed in this work is based on the extraction of the velocity map from each sensor unit through the optical flow computation [6]. For that, only left cameras of each stereo pair are enabled *(front, back, right, left)* in order to return grey images of the world in approximate 360°. However, even though we have four cameras observing the world at the same time, only one of these is used for localization *(main view)* in order to reduce computational times. The rest of the cameras are used as a backup in case the main view is obstructed. Obstructions are detected by the algorithm proposed in Sect. 4.1 and the way the next best view is selected is depicted in Sect. 4.2.

4.1 Camera-Blocking Detection

Let $C_i \in [front, back, right, left]$ with $i \in [1, 2, 3, 4]$ be the four different view directions taken from *Guidance*. For each C_i the optical flow O_{flow} is computed at every consecutive pair of frames f_i and f_{i+1}. After that, a set of ORB features [17] are computed and filtered by a threshold V_{min} (see Eq. 1). We choose ORB features because these are basically a fusion of FAST keypoints detector [16] and BRIEF descriptors [3] with several modifications to enhance the performance.

$$\mathbf{card}(\mathcal{F}^j)_i = \left| x \in F_i^j : \|O_{flow}(x)\| < V_{min} \right| \tag{1}$$

In Eq. 1, $\mathbf{card}(\cdot)$ represents the cardinality of the set $\mathcal{F}^j$, $\mathcal{F}^j$ is the set of all features such that their velocity are less than the threshold V_{min} *(static features)*, x is a feature computed by the ORB extractor, F_i^j represents the set of features computed in the j^{th} frame of the i^{th} camera, $||O_{flow}(x)||$ is the magnitude of the optical flow at x and V_{min} represents the maximum velocity for which a feature is considered static (see Fig. 2).

For every camera direction C_i a queue Q_i is created and filled with the number of static features of each frame ($\mathcal{F}_i^j$). If the size of the Q_i queue is equal to a given number of frames *(NF)* the older value of Q_i is released for storing a new one. The number NF is directly related with the time window to be analyzed and the frequency of the video device. For instance, if the sensor have a frequency of 18 fps and you want to store the last 2 s, then you have to set $NF = 36$. In other words, $NF = \text{fps} \times (seconds\_to\_store)$.

$$Q_i = \left\{ \mathbf{card}(\mathcal{F}_i^0),\ \mathbf{card}(\mathcal{F}_i^1),\ ...,\ \mathbf{card}(\mathcal{F}_i^{NF}) \right\} \tag{2}$$

In order to perform the obstacle detection, a frequency analysis over each Q_i is done and a 1D histogram, H_i, is constructed with the values of each Q_i

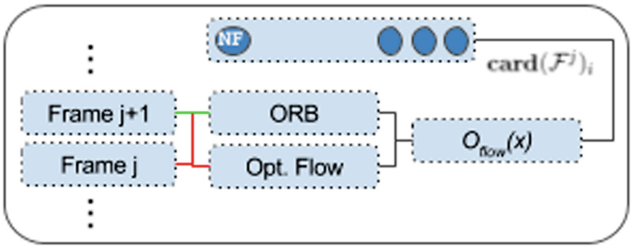


Fig. 2. In this picture $frame_j$ and $frame_{j+1}$ represent any consecutive pair of frames used for computing the optical flow O_{flow}. Over the $frame_j$ a set of ORB features are extracted and its velocities are computed using O_{flow} in order to extract the number of static features in $frame_j$. Finally, the number of static features per frame, $\mathbf{card}(frame_j)$, is stored in the queue of its respective camera.

once the size of Q_i reaches NF. Once the queues are filled for first time, the histograms computation is performed at each frame. The idea of using histograms for counting the frequency of the values in the queue allows us to determine if an obstacle is blocking temporally the main view or not. For instance, if a person walks in front of the main camera and then stops so that the main view gets blocked, then the number of static features will start to decrease at every frame and hence the queue of the main camera's view will start to have many values near to 0. This situation produces a transformation in the histogram where the first bin will become in the biggest bin within the histogram. However, when the person starts to move far from the camera view, the number of static features per frame will increase and consequently the queue values. This another situation produces that the last histogram's bins being the largest. Finally, a camera change can be made at the moment in which all the values of the Q_i are in the first bin of the H_i.

4.2 Camera Selection

At this point of the algorithm, the main camera can be considered blocked and the system has to evaluate the other views in order to select the best one of the rest to continue localized inside the world. For this case, the best view is such in which its Q_i contains the largest amount of static features over a long period of time [4].

As mentioned before in Sect. 4.1, if all values of the main view's histogram are in its first bin, then it is not longer convenient to keep viewing in that direction. Following the proposed in [4] a new good view is such a view that conserves more static features over a time interval. Therefore, the new best camera's view to stay localized is such that its histogram contains the highest statistical mode (see Sect. 5.2).

5 Experiments

The experiments presented in this section describe the relation between cameras under specific situations as well as the frequency analysis of the static features within the scenes. For the sake of a better understanding of this model we divide the experimental section in static and dynamic testing.

5.1 Static Test

The goal in this experiment is to assess the performance of our camera selection approach. For that, the number of bins per histograms is set empirically to 5, $NF = 36$ and the upper left corner image in Fig. 3(a) is selected as main view. As mentioned before, the number of static features is computed *(points in blue)* over the four different directions in order to generate the queues and later the histograms. Once the system is running, we proceed to block the main view as is shown in Fig. 3(b) and the system is able to select the best of the remaining views. In this case, if we look at the Fig. 3(b), we might realize that the next best view will be the one at the bottom right because in that view the number of static features is highest during a period of time.

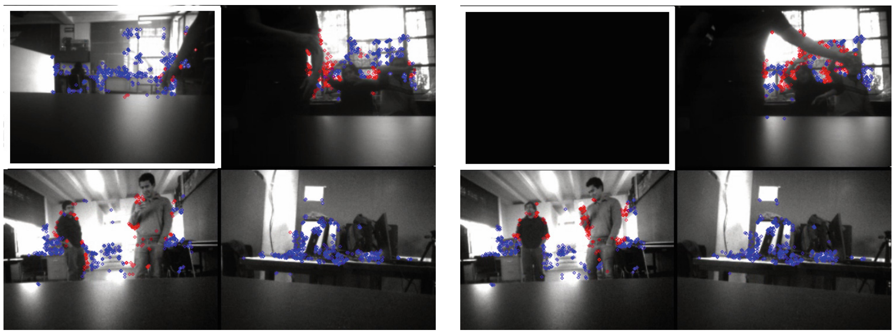
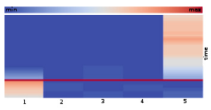
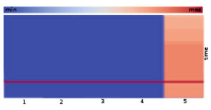
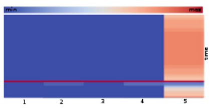

(a) Camera initialization. (b) Camera blocking.

Fig. 3. (a) Initial state of the system and the main view is enclosed in a white rectangle. (b) Static and non-static features represented in blue and red color respectively. (Best seen in color)

As we have explained before in Sect. 4, our camera selection approach is based on the frequency analysis of the static features behavior. For that, a 1D histogram is built for each view and a continuously evaluation is performed in order to see if the number of static features in the main view is lower than a threshold; if this happens then the main view has been blocked. Figure 4 shows the histogram behavior over the time for the four views. At the beginning, the main view's histogram (see Fig. 4(a)) has all its values in the last bin as well as the other views, however, there is an instant marked with a red line where


(a) Top left


(b) Top right


(c) Bottom left

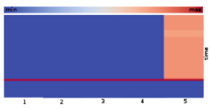

(d) Bottom right

Fig. 4. Histogram behavior over time. The red horizontal line represents the moment in which a change of the main view has happened. The bar at the top of each histogram is used to represent the frequency in the histogram, higher values are in red while lower values appears in blue. The time sequence starts at the top. The bin number is shown at the horizontal axis. (Best seen in color)

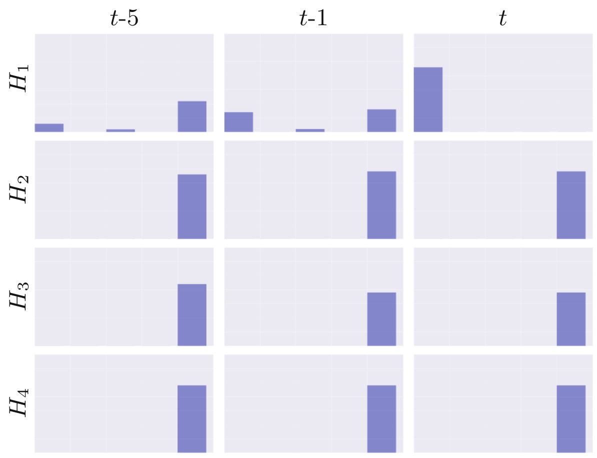


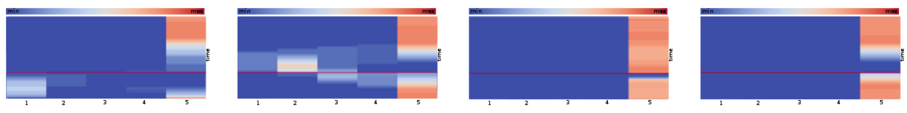
Fig. 5. Histograms evolution near the switching point. $H1$ represents the main view's histogram while $H2, H3$ and $H4$ the rest. Histograms are shown at frames t *(the switching point)*, $t-1$ and $t-5$. Note how the values in the histogram $H1$ move from the last bin to the first one from $t-5$ to t. For this case the selected camera is C_4 because numerically this histogram keep the highest mode during the last 2 s.

the histogram distribution starts to change, this is because the camera is being blocked, and therefore, the first bin starts to grow. At this point the system realizes that the camera was blocked and selects from the remaining views, the best one to stay localized. Finally, the system selects the bottom right view (see Fig. 3(b)) because as can be observed in Fig. 4(d), is the view where the histogram have the larger statistical mode *(darkest red color)*. See Fig. 5 for a better understnding of the histogram behavior near the switching point.

In Fig. 4(d) after switching cameras *(red line)* appears a period of time with no static features. This situation is presented because the camera was blocked with a human hand which introduces a low motion over the sensor and hence a perturbation in the optical flow.

5.2 Dynamic Test

For this experiment, our robot system is navigating in an indoor scenario. The robot is moving with the *Guidance* attached at the top of it (see Fig. 1(b)) and the *front* camera is designated for staying localized by using ORB-SLAM. While robot is navigating all features velocities are affected by the displacement vector. There are always two views that are more affected than the others. The *front* and *back* views are less affected because the motion is normal to both planes while *left* and *right* views depend strictly on the displacement vector. For instance, if the robot is moving forward, the velocities of each feature from the *front* and *back* views will not be affected because the optical flow won't be perturbed. On the other hand, for lateral views, almost all the feature won't be statics, this effect is like if the robot was static and the world was in motion.



(a) Top left (b) Top right (c) Bottom left (d) Bottom right

Fig. 6. Histogram behavior over the time when the sensor is mounted on the robot. The red horizontal line represents the moment in which a change of the main view has happened. (Best seen in color)

As in the first experiment, in Fig. 6 we depict the histograms behavior over the time. In this experiment the robot is initially static, hence, at the begging the amount of static features is high in each view. Once the robot starts moving the number of features starts to change. In this case it is more clear how the number of features on the main view (see Fig. 6(a)) starts to decrease while the robot is moving. Besides, in Fig. 6(c) it is possible to observe that the robot motion does not have effected the number of static features for this view, as we already mention above, this is because the motion is normal to this image plane. Finally, due to this effect, this view is the one that was selected as the new main view.

For lateral views (see Fig. 6(b) and (d)), we can distinguish a smooth transition in the last bin of the histograms which goes from orange to blue *(above to below)*. The above portion of the histograms, where the orange color remains constant is because we started to acquire data with the robot in an static state. In the histograms we can detect the starting robot motion when the orange color starts to turn in blue. Color changes in histograms implies that, at this time, the number of features with low velocities starts to decrease. This situation is presented because static features are apparently moving in the opposite direction that robot does.

The red line in Fig. 6 represents a stop in the robot motion produced by an obstruction in the current main view. At this point, the robot remains static during 2 s *(36 frames)* for acquiring the temporal distribution of the static ORB-features at each direction. With this tiny stop, the optical flow is finally stabilized and hence the features velocities.

6 Conclusions

In this paper we have presented a simple and effective camera selection algorithm for a multi-camera sensor systems when autonomous navigation is performed. Our approach aims at exploiting the visual capabilities offered by multi-camera sensors for visual-based localization, in scenarios where visual localization is obstructed. Experiments were conducted in static and dynamic scenarios. For the dynamic scenarios, we used a service robot platform *(Robot Sabina)*. In both cases, the results exhibit the same relation; the next best view is the one with more static features during a period of time, according to [4]. In addition, the proposed algorithm enables the system to maintain localization whilst keeping a low computational cost.

As a future work we are interested in incorporating the estimated displacement vector in order to reduce the perturbation at each view direction, specially in lateral views. In order to do that, we can incorporate another inertial measurement unit *(IMU)* and fuse its information with that of the robot odometry in order to enhance the selection of the best next main view.

Acknowledgments. This work was supported in part by FONCICYT (CONACYT and European Union) Project SmartSDK - No. 272727. Reinier Oves García is supported by a CONACYT Scholarship No.789638. Dr. J. Martinez-Carranza is thankful for the support received through the Newton Advanced Fellowship with reference NA140454.

References

1. Bapat, A., Dunn, E., Frahm, J.-M.: Towards kilo-hertz 6-DoF visual tracking using an egocentric cluster of rolling shutter cameras. IEEE Trans. Vis. Comput. Graph. **22**(11), 2358–2367 (2016)
2. Barbosa, M., Bernardino, A., Figueira, D., Gaspar, J., Gonçalves, N., Lima, P.U., Moreno, P., Pahliani, A., Santos-Victor, J., Spaan, M.T.J., Sequeira, J.: ISRobotNet: a testbed for sensor and robot network systems. In: IROS, pp. 2827–2833 (2009)
3. Calonder, M., Lepetit, V., Strecha, C., Fua, P.: BRIEF: binary robust independent elementary features. In: Daniilidis, K., Maragos, P., Paragios, N. (eds.) ECCV 2010. LNCS, vol. 6314, pp. 778–792. Springer, Heidelberg (2010). https://doi.org/10.1007/978-3-642-15561-1_56
4. Costante, G., Forster, C., Delmerico, J.A., Valigi, P., Scaramuzza, D.: Perception-aware path planning. IEEE Trans. Robot. (2016). arXiv preprint arXiv:1605.04151
5. Das, A., Waslander, S.L.: Entropy based keyframe selection for multi-camera visual slam. In: 2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), pp. 3676–3681. IEEE (2015)

6. Farnebäck, G.: Two-frame motion estimation based on polynomial expansion. In: Bigun, J., Gustavsson, T. (eds.) SCIA 2003. LNCS, vol. 2749, pp. 363–370. Springer, Heidelberg (2003). https://doi.org/10.1007/3-540-45103-X_50
7. Forster, C., Pizzoli, M., Scaramuzza, D.: SVO: fast semi-direct monocular visual odometry. In: 2014 IEEE International Conference on Robotics and Automation (ICRA), pp. 15–22, May 2014
8. Guyue, Z., Lu, F., Ketan, T., Honghui, Z., Kai, W., Kang, Y.: Guidance: a visual sensing platform for robotic applications. In: 2015 IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW), pp. 9–14, June 2015
9. Harmat, A., Trentini, M., Sharf, I.: Multi-camera tracking and mapping for unmanned aerial vehicles in unstructured environments. J. Intell. Robot. Syst. **78**(2), 291–317 (2015)
10. Houben, S., Quenzel, J., Krombach, N., Behnke, S.: Efficient multi-camera visual-inertial slam for micro aerial vehicles. In: IROS 2016, pp. 1616–1622. IEEE (2016)
11. Kaess, M., Dellaert, F.: Probabilistic structure matching for visual SLAM with a multi-camera rig. Comput. Vis. Image Underst. **114**(2), 286–296 (2010). Special issue on Omnidirectional Vision, Camera Networks and Non-conventional Cameras
12. Kazik, T., Kneip, L., Nikolic, J., Pollefeys, M., Siegwart, R.: Real-time 6D stereo visual odometry with non-overlapping fields of view. In: 2012 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 1529–1536. IEEE (2012)
13. Klein, G., Murray, D.: Parallel tracking and mapping for small AR workspaces. In: Mixed and Augmented Reality, ISMAR 2007, pp. 225–234. IEEE (2007)
14. Mur-Artal, R., Tardós, J.D.: ORB-SLAM2: an open-source slam system for monocular, stereo, and RGB-D cameras. IEEE Trans. Robot. **33**(5), 1255–1262 (2017)
15. Netramai, C., Roth, H., Sachenko, A.: High accuracy visual odometry using multi-camera systems. In: 2011 IEEE 6th International Conference on Intelligent Data Acquisition and Advanced Computing Systems (IDAACS), vol. 1, pp. 263–268. IEEE (2011)
16. Rosten, E., Drummond, T.: Machine learning for high-speed corner detection. In: Leonardis, A., Bischof, H., Pinz, A. (eds.) ECCV 2006. LNCS, vol. 3951, pp. 430–443. Springer, Heidelberg (2006). https://doi.org/10.1007/11744023_34
17. Rublee, E., Rabaud, V., Konolige, K., Bradski, G.: ORB: an efficient alternative to sift or surf. In: 2011 IEEE International Conference on Computer Vision (ICCV), pp. 2564–2571. IEEE (2011)
18. Tardif, J.-P., Pavlidis, Y., Daniilidis, K.: Monocular visual odometry in urban environments using an omnidirectional camera. In: IEEE/RSJ International Conference on Intelligent Robots and Systems, IROS 2008, pp. 2531–2538. IEEE (2008)
19. Zhou, G., Fang, L., Tang, K., Zhang, H., Wang, K., Yang, K.: Guidance: a visual sensing platform for robotic applications. In: CVPR, pp. 9–14 (2015)