26th International Congress of Mechanical Engineering
November 22-26, 2021, Virtual Congress, Brazil

COB-2021-1051
EFFICACY TEST OF THE VISUAL ODOMETRY METHOD ON TERRASENTIA ROBOT

Jorge Id Facuri Filho
Vitor Akihiro Hisano Higuti
University of São Paulo, Av Trabalhador São-Carlense, 400, São Carlos, São Paulo, Brazil
jorgeid@usp.br, vitor.higuti@usp.br

Marcelo Becker
University of São Paulo, Av Trabalhador São-Carlense, 400, São Carlos, São Paulo, Brazil
becker@sc.usp.br

Abstract. *In the agricultural scenario, some of the biggest challenges that farmers suffer are the identification of trails, navigation and mapping of remote and hard to access regions. In this context, visual odometry is one of the methods to obtain positioning and orientation information, which when applied to a mobile robot, removes the need for people to move around in these difficult access environments. Since it relies on a single monocular camera, visual odometry (VO) is often cheaper and more flexible in the robot assembly when compared to common localization sources such as GPS. Additionally, visual odometry recovers the robot's trajectory only by analyzing the changes of feature points in the captured images. Thus, a visual odometry algorithm is implemented using the images captured by TerraSentia robot to enable the estimation of its trajectory in an agricultural environment. FAST detector with the Kanade-Lucas-Tomasi tracker for extraction with tracking is implemented to compare the trajectory from VO with the trajectory from EKF data. In a scenario without interferences that disturb the tracker, only drift errors are present in the estimated trajectory. The path recovered by VO gets to follow the ground truth, with less than 1 m of translation error.*

Keywords: *Mobile Robot, Visual Odometry, Agriculture*

1. INTRODUCTION

Since the Neolithic period, the discovery of agricultural activities provided man the possibility of transitioning from the nomadic lifestyle to the sedentary lifestyle. In this sense, agriculture enabled the stabilization of man in space, becoming the main source of food production currently. This factor strengthens the socioeconomic level of countries and fights hunger (PRABURAJ, 2018). Linked to this great importance, the main technologies developed nowadays aim to intensify agricultural processes and accelerate them. The use of mobile robots is one of the most appropriate technologies for these goals. To optimize farmers' tasks, mobile robots can perform some activities in the agricultural field such as mapping regions, identifying trails and remote areas, automatic seed planting, automated harvesting and self-navigation. (ROBOTICS ONLINE MARKETING TEAM, 2019).

The Mobile Robotics Laboratory (LabRoM) of São Carlos School of Engineering (EESC) of USP, in partnership with Illinois University, are studying visual odometry (VO) method to implement in TerraSentia robot. TerraSentia is an agricultural mobile robot that goes under canopy in row patterned fields. VO is an algorithm than can be applied in a mobile robot to facilitate the tasks in the agricultural field. It is a method for obtaining positioning and orientation information from the robot. It is applied to determine robot's location in a navigable agricultural region that has narrow trails and is full of plants around. Visual odometry works only by analysing the changes of features in a sequence of frames, captured by a camera (Fraundorfer and Scaramuzza, 2012). The knowledge of the features changes in the scenario enables the motion estimation and the robot trajectory recovery (Scaramuzza and Fraundorfer, 2011).

Studies about the capable of mobile robots determining their own location in the environment, has been performed since the mid-1980s. Initial studies of VO were started by NASA (National Aeronautics and Space Administration), with the aim of applying the method in autonomous robots in the Mars explorations missions (Cheng *et al.*, 2005). VO has some advantages compared to traditional sources for obtaining positioning, as GPS (Global Positioning System) and encoders (ZAMAN *et al.*, 2019). Compared to the GPS, VO is cheaper, more flexible in the robot assembly and does not suffer the problem of signal drop that causes poor accuracy in providing position information. Compared to the encoders, VO is susceptible to a lower error rate in sliding environments, despite the existence of drift errors that can be minimized.

2. ROBOT CAMERA SETUP

In this work, environment data to get the VO results were collected by a monocular camera of TerraSentia robot (2020 version). Figure 1 shows TerraSentia robot and its model of camera that is a 2.0 USB 5Mp OMNIVISION OV5640 color CMOS sensor 2.1MM perspective camera (Ailipu Technology Co. Ltd, 2014). TerraSentia robot is an autonomous robot developed at the University of Illinois, that was provided to LabRoM. The robot was built for researchers make new discoveries on navigation technologies in the agricultural field. Originally, the nominal height of TerraSentia camera to the ground is 0.24 m with an uncertainty of $\pm$ 0.01 m, because of robot's suspension.

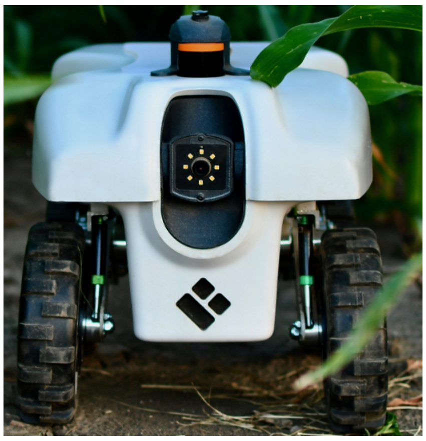

Figure 1. TerraSentia robot and its camera. Retrieved from Earth Sense (2020).

3. VO ALGORITHM APPROACH

VO implemented in this work is the algorithm from 2d-2d correspondences, proposed by Scaramuzza and Fraundorfer (2011) and shown in Alg. 1. Basically in this approach, a set of frames $I_{0:n} = \{I_0, ..., I_n\}$ is taken at times k from a monocular camera and features f_{k-1} and f_k from two consecutive frames are extracted. Then, features in frame I_{k-1} are matched with features in frame I_k or tracked in frame I_k to motion estimation. Between these last two steps, a matrix $\mathbf{E}_{k,k-1}$, named essential matrix, is computed between two consecutive frames in the robot's movement. The decomposition of $\mathbf{E}_{k,k-1}$ allows us to recover rotation matrix $\mathbf{R}_{k,k-1}$ and translation vector $\mathbf{t}_{k,k-1}$ of relative robot camera motions, enabling full trajectory recovery. This work used the tracking method to get features correspondences. According to Fraundorfer and Scaramuzza (2012), detect and track approach is useful in applications where amount of motion in two consecutive frames are small. In a agricultural scenario most part of features and texture changes in the environment are almost homogeneous, making the tracking method work well in small image sequences. Each step of VO algorithm implemented will be shown in next subsections, in details.

Algorithm 1 VO from 2-D-to-2-D correspondences

```
1. Capture a frame $I_{k-1}$;
2. Extract features in frame $I_{k-1}$;
3. Track features of $I_{k-1}$ in $I_k$;
4. Compute essential matrix for image pair $I_{k-1}$ and $I_k$;
5. Recover rotation matrix $\mathbf{R}_{k,k-1}$ and translation vector $\mathbf{t}_{k,k-1}$ between two camera positions, by $\mathbf{E}_{k,k-1}$ decomposition;
6. Recover trajectory by motion estimation with elements of item 5;
7. Repeat from item 1).
```

3.1 Frame capture

First, in frame capture process, it is necessary to know the projection model of camera to get the set of images $I_{0:n}$ in each time k. This knowledge is used to know how to configure the camera to correctly work. In this work, the camera in the robot setup is a monocular perspective camera, then the pinhole model is used. According to Zisserman (2004), pinhole model is a camera model that relate a point in 3D space with its projection in an image plane. Let $\mathbf{X} = [\mathrm{X}, \mathrm{Y}, \mathrm{Z}]^T$ be a point in space, related to camera coordinates, f is the focal length of camera and image plane Z = f. The mapping approach from Euclidean 3-space $\mathbb{R}^3$ to Euclidean 2-space $\mathbb{R}^2$ of pinhole model is explained by Fig. 2.

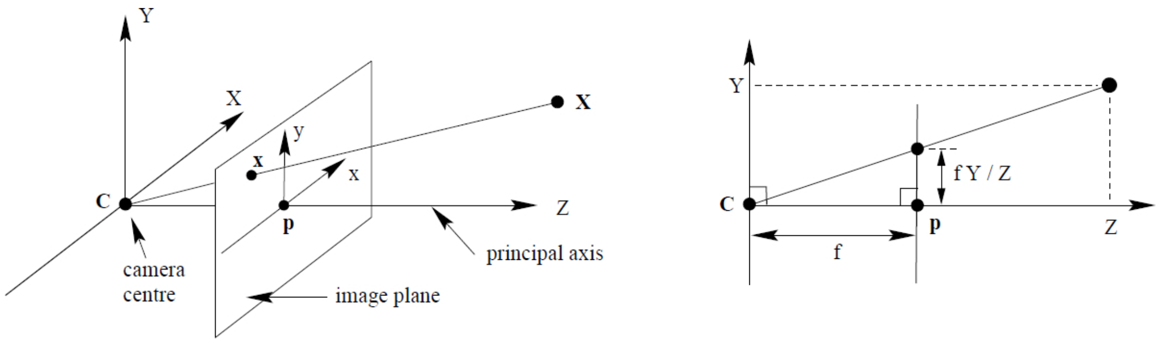

Figure 2. Camera pinhole model. Retrieved from Zisserman (2004).

3.1.1 Camera Calibration

Another important step of frame capture is the camera calibration. Camera calibration is a method to obtain the intrinsic parameters of camera, like focal length (f), image coordinate of optical center (o_x and o_y) and effective size of the pixels (s_x and s_y). According to Trucco and Verri (1998), the intrinsic parameters are responsible for relating the coordinate system of camera with the coordinate system of image, in pixel. Also, these parameters are used to get a calibration matrix **K**, shown in Eq. (1). Some tools like Camera Calibration toolbox for Matlab, proposed by Jean-Yves Bouguet (2015), and a C implementation with OpenCV library (OpenCV, 2021), can be used to calibrate perspective cameras and get **K** parameters.

$$\mathbf{K} = \begin{bmatrix} \frac{\text{-f}}{s_x} & 0 & o_x \\ 0 & \frac{\text{-f}}{s_y} & o_y \\ 0 & 0 & 1 \end{bmatrix} \tag{1}$$

3.2 Features extraction

One of the most important step of VO is feature extraction. This step is responsible to get relevant information about the environment in each captured frame. The extracted features, along two consecutive frames, allow the VO algorithm to compare the scenes changes and subsequently estimate robot's trajectory (Scaramuzza and Fraundorfer, 2011). Some types of features that can appear in images are corners, that represent the intersection of two edges, and blobs that is an image pattern. According to Fraundorfer and Scaramuzza (2012), corner features are faster and less distinctive to compute than blob features. Thus, they are better to use in agricultural environments once textures are most homogeneous. There are many types of corner detectors, such as Harris Corner Detector (Harris *et al.*, 1988), Shi Tomasi (Shi *et al.*, 1994) and Fast Detector, that can be applied in different types and approaches of VO. Işık (2014) reached good results about the number of corners detected and time processing of detection of FAST detector, which motivated this work to choose FAST detector algorithm for VO implementation.

FAST Corner Detector proposed by Rosten and Drummond (2006) is a detector that makes a segmentation test around a specific point p, centered in a 16 pixel circle, adopted to be a possible corner. Then, to verify if the corner is real, a set n of adjacent points of p is chosen and its intensity is analysed. If the intensity is clearer than $(I_p + thr)$, or darker than $(I_p - thr)$, where I_p is the intensity of p and thr is a threshold parameter of FAST, adopted corner is a real corner. This process is denominated adjacency criterion. A faster test can be applied analysing 1, 5, 9 and 13 pixels position, according to Fig. 3. The adjacency criterion can only be applied if three of these pixels position have a intensity clearer than $(I_p + thr)$ or darker than $(I_p - thr)$.

3.3 Features Tracking

For features tracking, Kanade-Lucas-Tomasi (KLT) can be applied in VO problem to track features f_{k-1} of frame I_{k-1} in frame I_k. Bouguet *et al.* (2001) proposed a method that considers two consecutive images I and J, in a grayscale, and their intensities $I(\mathbf{u})$ and $J(\mathbf{v})$ at the $\mathbf{u}$ and $\mathbf{v}$ points. Let $\mathbf{u} = (u_x, u_y)^T$, the KLT process targets finding $\mathbf{v}$ that satisfies Eq. (2).

$$\mathbf{v} = \mathbf{u} + \mathbf{d} = (u_x + d_x, u_y + d_y)^T, \tag{2}$$

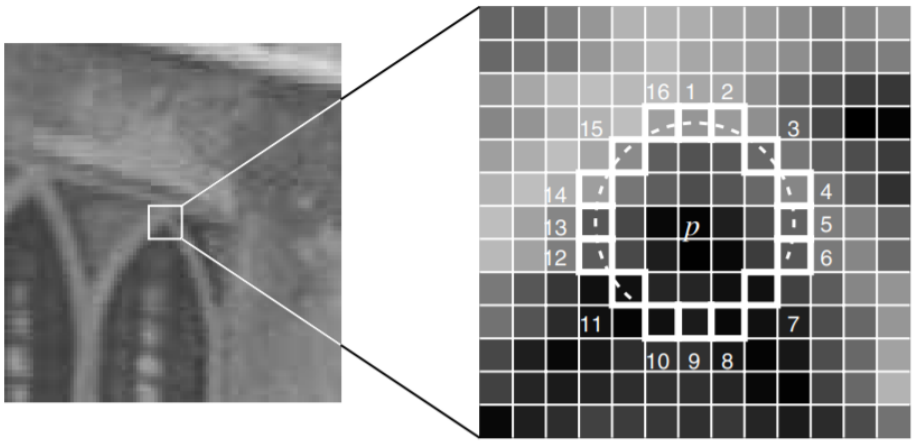

Figure 3. Demonstration of FAST corner detector 16 pixel segmentation test. Retrieved from Rosten and Drummond (2006).

where $\mathbf{d} = (dx, dy)^T$ is the optical flow that represents the displacement of **u** point between the two images. Because KLT works well at small motions and fails at large motion, Bouguet *et al.* (2001) introduces the pyramidal level representation concept to solve this problem. This representation guarantees a trade-off relation between robustness and precision of the tracker to compute motions.

Basically, KLT approach reduces image resolution in a scene from an original image, in a pyramidal form, i.e, in a recursive way. It means that pyramid level L increases while image resolution decreases. Let L_m be the maximum level of pyramid and I^0 be the image in original resolution, thus the pyramidal method will build image I^1 from I^0, I^2 from I^1 and so on until I^{Lm}. To find Eq. (2), in terms of pyramidal level approach, it is necessary to find the optical flow $\mathbf{d}^L$ that minimizes the residual function ϵ^L described by Eq. (3).

$$\epsilon^L(\mathbf{d}^L) = \epsilon^L(d_x^L, d_y^L) = \sum_{x=u_x^L-w_x}^{u_x^L+w_x} \sum_{y=u_y^L-w_y}^{u_y^L+w_y} \left[I^L(x,y) - J^L(x+g_x^L+d_x^L, y+g_y^L+d_y^L) \right]^2, \tag{3}$$

where $\mathbf{g}^L = (g_x^L, g_y^L)^T$ is an initial guess for optical flow at level L, w_x and w_y are the size of an integration window $(2w_x + 1)$ x $(2w_y + 1)$ around each point $\mathbf{u}^L$.

3.4 Essential matrix computation

After features detection and tracking, it is necessary to compute the essential matrix **E**. This matrix describes the geometric relations between two frames, captured by a calibrated camera (Scaramuzza and Fraundorfer, 2011). To obtain **E**, Five-Point algorithm proposed by Nistér (2004) is used. This method chooses at least five 2d-2d correspondent points $\mathbf{q} = (u, v, 1)$ and $\mathbf{q}' = (u', v', 1)$, that respect the epipolar constraint defined by Eq. (4), to find **E** between two consecutive frames. In practice, a different essential matrix $\mathbf{E}_{k,k-1}$ is computed for every two consecutive frames in VO problem. This occurs because features in a set of frames are dynamic and change along the trajectory.

$$\mathbf{q}'\mathbf{E}\mathbf{q} = 0 \tag{4}$$

To improve VO results, it is necessary to apply with Five-Point algorithm the RANSAC method (Fischler and Bolles, 1981). The set of tracked features has many outliers, and their presence in the motion estimation will result in translation and rotation errors. Thus, RANSAC is a probabilistic method that eliminates outliers to estimate motion model more accurately. Algorithm 2 explains RANSAC method approach, according to Fraundorfer and Scaramuzza (2012).

Equation (5) shows how to determine the number of iterations N of RANSAC method.

$$N = \frac{log(1-p)}{log(1-(1-\epsilon)^s)}, \tag{5}$$

where p is the probability of success required by the method and ϵ is the percentage of outliers.

Algorithm 2 RANSAC method

```
1. Consider a set $A$ of tracked features;
2. Randomly select a sample points $s$ of $A$;
3. Fit a model to the sample $s$ (in the case of the VO application, fit the model of relative motion between two images);
4. Compute the distance $d$ from all points in the sample to the built model;
5. Build the set of inliers, whose distance to the model is smaller than the distance $d$;
6. Store these points that respect the distance condition;
7. Repeat process 1 to 6 up to a maximum number of iterations $N$;
8. From all the iterations performed, choose the set that has the highest number of inliers as a solution to the problem;
9. Estimate the model using all the selected points.
```

3.5 Essential Matrix Decomposition

Essential Matrix also can be defined multiplying a rotation matrix by a translation vector, between every two consecutive camera positions (Scaramuzza and Fraundorfer, 2011). Equation (6) shows this relation.

$$\mathbf{E}_{k,k-1} = \hat{\mathbf{t}}_{k,k-1}\mathbf{R}_{k,k-1} \Rightarrow \mathbf{E_{k,k-1}} = \begin{bmatrix} 0 & -t_z & t_y \\ t_z & 0 & -t_x \\ -t_y & t_x & 0 \end{bmatrix} \mathbf{R}_{k,k-1}, \tag{6}$$

where $\mathbf{t}_{k,k-1} = [t_x, t_y, t_z]^T$. In Eq. (6), $\mathbf{R}_{k,k-1}$ and $\hat{\mathbf{t}}_{k,k-1}$ are the rotation matrix and the symmetric matrix of translation vector $\mathbf{t}_{k,k-1}$ components, respectively, of frame I_k with respect to frame I_{k-1}. From $\mathbf{E}_{k,k-1}$, obtained by Five-Point Algorithm, the VO problem will extract the rotation matrix and translation vector. To extract them, Singular Value Decomposition (SVD) can be applied to decompose $\mathbf{E}$, as in Eq. (7) (Scaramuzza and Fraundorfer, 2011). Finally, rotation matrix and translation vector can be found by Eq. (8) and Eq. (9), respectively.

$$\mathbf{E} = \mathbf{U}\mathbf{S}\mathbf{V}^T \tag{7}$$

$$\mathbf{R} = \mathbf{U}(\pm\mathbf{W}^T)\mathbf{V}^T \tag{8}$$

$$\hat{\mathbf{t}} = \mathbf{U}(\pm\mathbf{W})\mathbf{S}\mathbf{U}^T, \tag{9}$$

where,

$$\mathbf{W} = \begin{bmatrix} 0 & \pm 1 & 0 \\ \mp 1 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{10}$$

Analysing Eq. (8) and Eq. (9), it is possible to conclude that there are four possible solutions for $\mathbf{R}$ and $\mathbf{t}$ for each $\mathbf{E}$ estimated. Triangulation (Zisserman, 2004) can be applied to find the correct solution for $\mathbf{R}$ and $\mathbf{t}$.

3.6 Motion Estimation

The extraction of rotation matrix and translation vector allows to build the rigid body transformation matrix $\mathbf{T}_{k,k-1}$. $\mathbf{T}_{k,k-1}$ matrix, shown in Eq. (11), describes the robot camera motion at time k with respect to the motion at time $k-1$ (Scaramuzza and Fraundorfer, 2011). For simplicity, the camera coordinate system of the robot is assumed to be the same as the robot.

$$\mathbf{T}_{k,k-1} = \begin{bmatrix} \mathbf{R}_{k,k-1} & \mathbf{t}_{k,k-1} \\ 0 & 1 \end{bmatrix} \tag{11}$$

Computing the rigid body transformation of the camera motion at other times, it is possible to get the set $\mathbf{T}_{1:n} = \{\mathbf{T}_{1,0}, ..., \mathbf{T}_{n,n-1}\}$. $\mathbf{T}_{1:n}$ contains all the relative camera motions along the path traveled by robot. Furthermore, the rigid body transformation set allows to compute the camera pose $\mathbf{C}_{k,0}$ at time k with respect to the camera pose at time $k = 0$. $\mathbf{C}_{0:n} = \{\mathbf{C}_0, \mathbf{C}_{1,0}, ..., \mathbf{C}_{n,0}\}$ represents the set of camera poses, which $\mathbf{C}_0$ is the camera pose at time $k = 0$. $\mathbf{C}_0$ can be determined arbitrarily. Assuming $\mathbf{C}_0$ as in Eq. (12), the camera pose at time k is found concatenating the camera poses at previous times, as shown in Eq. (13).

$$\mathbf{C}_0 = \begin{bmatrix} \mathbf{R}_{0,0} & \mathbf{t}_{0,0} \\ 0 & 1 \end{bmatrix} = \begin{bmatrix} \mathbf{I}_{3x3} & 0 \\ 0 & 1 \end{bmatrix} \tag{12}$$

$$\mathbf{C}_{k,0} = \mathbf{C}_{k-1,0}\mathbf{T}_{k,k-1} \Rightarrow \mathbf{C}_{k,0} = \begin{bmatrix} \mathbf{R}_{k-1,0}\mathbf{R}_{k,k-1} & \mathbf{R}_{k-1,0}\mathbf{t}_{k,k-1} + \mathbf{t}_{k-1,0} \\ 0 & 1 \end{bmatrix} \tag{13}$$

Equation (13) determines that translation vector and rotation matrix of camera pose $\mathbf{C}_{k,0}$ are $\mathbf{R}_{k-1,0}\mathbf{t}_{k,k-1} + \mathbf{t}_{k-1,0}$ and $\mathbf{R}_{k-1,0}\mathbf{R}_{k,k-1}$, respectively. The obtained set of camera poses is sufficient to get motion estimation and recover robot's trajectory. It is import to emphasize that $\mathbf{E}_{k,k-1}$ contains the camera motion parameters up to an unknown scale for translation vector. This scale must be found to compute camera poses correctly. Aqel *et al.* (2016) shows some methods to solve the scale uncertainty problem of translation vector.

4. EXPERIMENT

The monocular VO algorithm was implemented on ROS framework using OpenCV library for C++ language, in Ubuntu 18.04 LTS. The algorithm ran in a laptop with NVIDIA GeForce GTX 1650 4 Gb video card, AMD Ryzen 5 processor and 8 Gb RAM. The output data of algorithm was processed at Matlab to get the VO results. The dataset used is a set of images of the path traveled by TerraSentia Robot in an agricultural field, along a 90 m distance, approximately. The path has an "U" format, i.e, the robot leaves one trail and goes to another, performing an "U" curve.

The dataset has 4582 images, but there are some frames that are rejected in the algorithm. This frames are rejected because of the presence of some plants that occludes camera vision (see Fig. 4), inducing more errors during motion estimation. For frames rejection, it was implemented a logic that computes the percentage of inliers of frame I_{k-1} tracked on frame I_k. If inliers percentage is less or equal than 65%, the frame at time k is skipped, the camera pose at this time is not considered, the features are redetected in frame I_{k+1} and the tracking process restarts. KLT tracker, used in the VO implementation, has a problem related to features loss during tracking process. To solve this problem and avoid the lack of enough features to motion estimation, the redetection logic was used every five frames. Table 1 shows the parameters of the methods used in each step of the implemented algorithm.

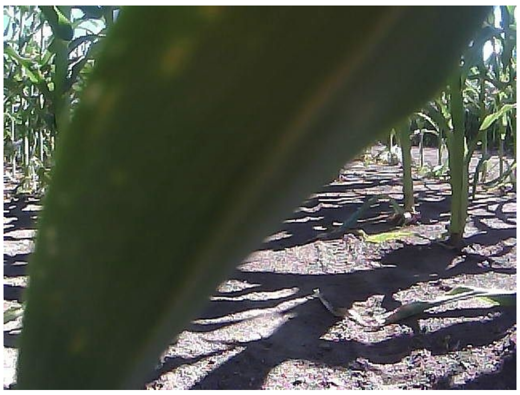

Figure 4. Example of the camera vision occlusion caused by the presence of plants on the trail.

Table 1. Parameters of the methods used in Monocular VO implementation.

Method	Parameters	
FAST Detector	$thr = 60$	-
KLT Tracker	Integration Window = 15 x 15	Level $L = 3$
RANSAC	$d = 1.0$	$p = 0.999$

TerraSentia's camera calibration matrix is shown in Eq. (14). The camera was calibrated from factory in 1280 x 720 resolution. The dataset used has images in 640 x 480 resolution, for this reason it was necessary to multiply x image coordinate parameters by $\frac{1}{2}$ and y coordinate parameters by $\frac{2}{3}$.

$$\mathbf{K} = \begin{bmatrix} 529.92 & 0 & 320.77 \\ 0 & 706.22 & 233.41 \\ 0 & 0 & 1 \end{bmatrix} \tag{14}$$

5. RESULTS

VO implemented was applied in three different situations using the dataset. The results were compared with EKF data, assumed as ground truth in the problem and the absolute scale of VO was obtained from EKF as reference. In the first situation (Fig. 5), the algorithm was applied in the whole dataset, considering the VO approach with the redetection logic. It was done to test how the algorithm would deal with long distances in the agricultural environment. But, in this test there was in the middle of the trail a interference, caused by a plant fallen on the ground. It was detrimental for VO experiment, because when TerraSentia moved over the plant the optical flow of KLT tracker identified the motion of the plant, caused by robot's passage, beyond the camera motion. Two different and independent motions in a same scenario may cause oscillations and errors in the tracker's operation. In the second situation (Fig. 6), VO was applied manually skipping the frames that has the plant fallen on the ground. This approach allowed algorithm to ignore that interference in scenario. In the third situation (Fig. 7), as initial position, VO was only applied after plant fallen on the ground interference.

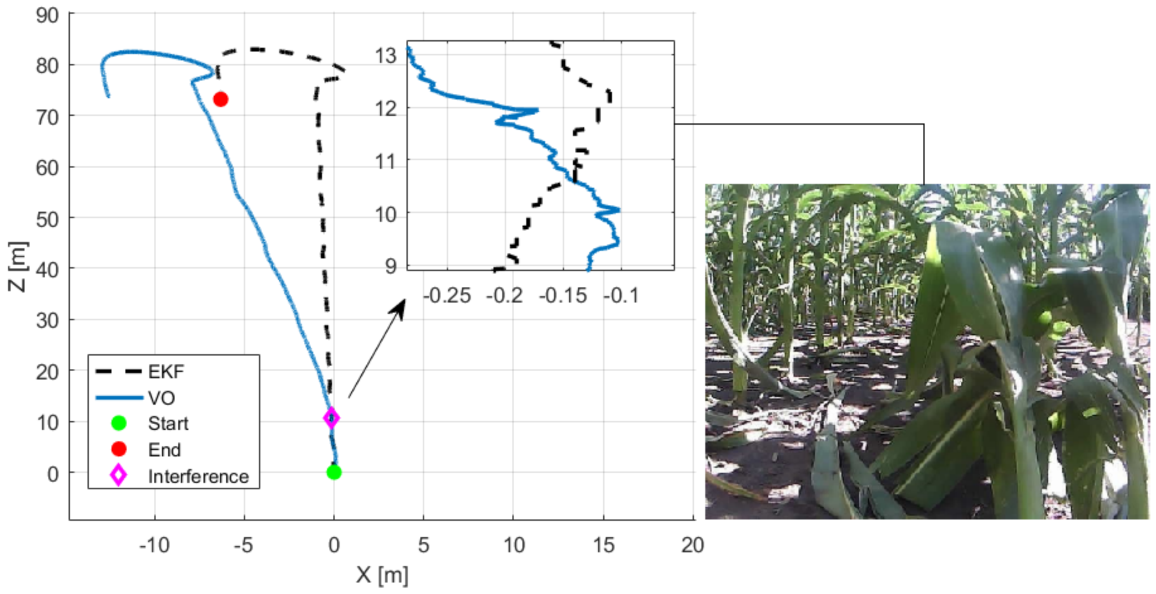

Figure 5. Comparison of monocular VO result with interference of plant fallen on the ground, in the middle of the trail, with EKF result (Situation 1).

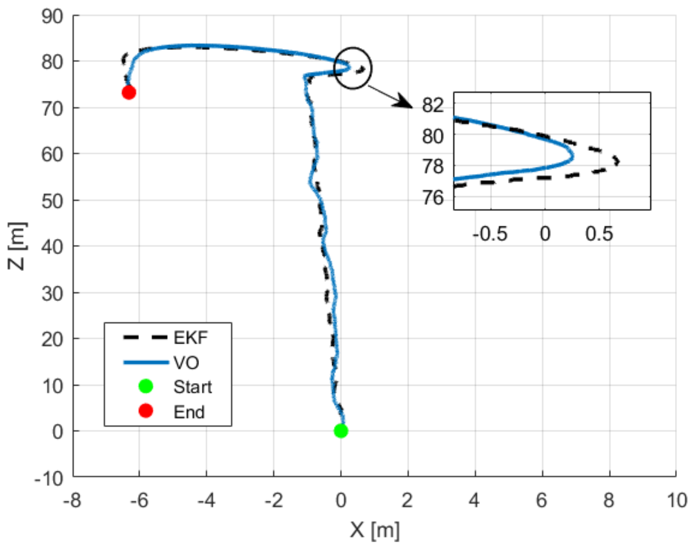

Figure 6. Comparison of monocular VO result skipping interference of plant fallen on the ground, in the middle of the trail, with EKF result (Situation 2).

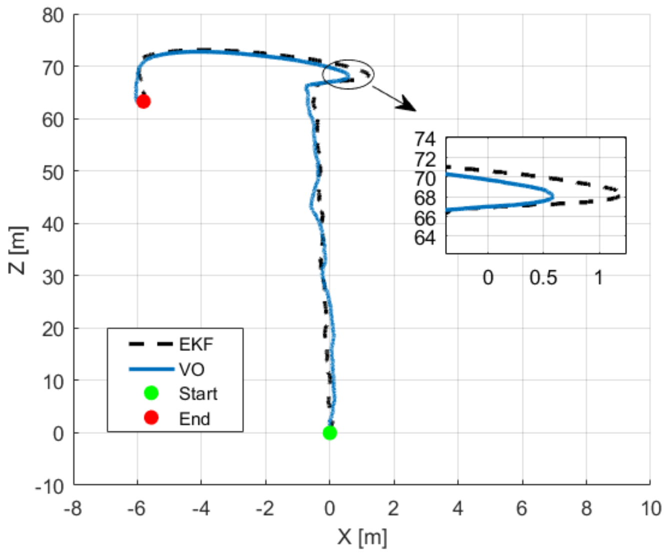

Figure 7. Comparison of monocular VO result starting after interference of plant fallen on the ground, in the middle of the trail, with EKF result (Situation 3).

Figure 5 shows that the interference caused by the plant on the ground, between 11.7 m and 12 m directly affected the VO trajectory. The excessive motion in the scene destabilized the tracker. Although the path of VO has deviated the trajectory, its format and distance became similar to the ground truth. Figure 6 shows that skipping frames with the interference, the trajectory of VO does not deviated abruptly and the estimated path got to follow the ground truth. VO got to estimate the curve done by TerraSentia to change the trail. Figure 7 shows similar results of VO to the previous situation. The drift problem of VO is evident in all the three situations, especially in the second and the third ones. This problem accumulates errors in the robot's translation, generating small oscillations over long straight line distances.

To evaluate the efficacy of VO result, four key points in the recovered path were chosen to verify the translation errors. Figure 8 shows the position of key points analyzed as reference. They were chosen considering the main parts of the path: the middle of the straight path, approximately, the start of the first curve, the start of the second curve and the end of the trajectory. Translation errors of each simulation are presented in Tab. 2, Tab. 3 and Tab. 4

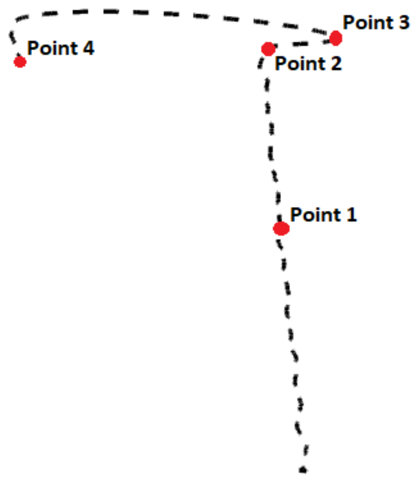

Figure 8. Key points chosen to evaluate translation errors in the trajectory.

Table 2. Translation errors of Situation 1.

Translation errors [m]	**Point 1**	**Point 2**	**Point 3**	**Point 4**
Error in X	2.52	6.97	7.45	6.22
Error in Z	0.31	0.37	0.44	0.6

Table 3. Translation errors of Situation 2.

Translation errors [m]	**Point 1**	**Point 2**	**Point 3**	**Point 4**
Error in X	0.087	0.14	0.42	0.037
Error in Z	0.05	0.33	0.33	0.8

Table 4. Translation errors of Situation 3.

Translation errors [m]	**Point 1**	**Point 2**	**Point 3**	**Point 4**
Error in X	0.15	0.36	0.62	0.17
Error in Z	0.16	0.23	0.05	0.05

Table 3 and 4 show that VO achieved a good precision in the recovered trajectories, with translation errors less than 1m. Table 2 shows that the interference on the ground caused big translation errors in X coordinates. When TerraSentia moved over the plant on the ground, the path deviated varied much more in X coordinate than Z coordinate, which ensured small errors in the vertical motion.

6. CONCLUSION

VO algorithm implemented got to recover the trajectory with a good accuracy when compared to EKF, assumed as ground truth. Translation errors less than 1 m were obtained in the situations which there were not interferences on the scene that disturb the tracker. KLT tracker got to perform well the tracking function, despite being very sensitive to interferences that add motion between frames, such as the fallen plant motion by robot's movement. The interference caused by fallen plant on the ground can greatly increase the translation error in the horizontal coordinate. This error could be minimized by applying computer vision algorithms to recognize fallen plants on the ground, making VO ignores the frames of the interference.

The redetection logic got to solve the problem of camera occlusion, caused by some plants, avoiding more error propagation during motion estimation because of the drift problem. VO approach is an incremental method to estimate pose, thus the small oscillations in the trajectory caused by the drift problem is inevitable.

7. ACKNOWLEDGEMENTS

This paper was supported by grants no. 2020/10533-0 and no. 2018/10894-2, São Paulo Research Foundation (FAPESP). The authors also would like to thank EESC-USP, for their support in this work.

8. REFERENCES

Ailipu Technology Co. Ltd, 2014. "5MEGAPIXEL USB CAMERA MODULE USB2.0 OMNIVISION OV5640 COLOR CMOS SENSOR 2.1MM LENS". `elpcctv.com/5megapixel-usb-camera-module-usb20-omnivision-ov5640-color-cmos-sensor-21mm-lens-p-215.html`. Accessed 02 June 2021.

Aqel, M.O., Marhaban, M.H., Saripan, M.I. and Ismail, N.B., 2016. "Review of visual odometry: types, approaches, challenges, and applications". *SpringerPlus*, Vol. 5, No. 1, pp. 1–26.

Bouguet, J.Y. *et al.*, 2001. "Pyramidal implementation of the affine lucas kanade feature tracker description of the algorithm". *Intel corporation*, Vol. 5, No. 1-10, p. 4.

Cheng, Y., Maimone, M. and Matthies, L., 2005. "Visual odometry on the mars exploration rovers". In *2005 IEEE International Conference on Systems, Man and Cybernetics*. IEEE, Vol. 1, pp. 903–910.

Earth Sense, 2020. "Earth Sense Agricultural Intelligence". `earthsense.co/`. Accessed 18 June 2021.

Fischler, M.A. and Bolles, R.C., 1981. "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.

Fraundorfer, F. and Scaramuzza, D., 2012. "Visual odometry: Part ii: Matching, robustness, optimization, and applications". *IEEE Robotics & Automation Magazine*, Vol. 19, No. 2, pp. 78–90.

Harris, C.G., Stephens, M. *et al.*, 1988. "A combined corner and edge detector." In *Alvey vision conference*. Citeseer, Vol. 15, pp. 10–5244.

Işık, Ş., 2014. "A comparative evaluation of well-known feature detectors and descriptors". *International Journal of Applied Mathematics Electronics and Computers*, Vol. 3, No. 1, pp. 1–6.

Jean-Yves Bouguet, 2015. "Camera Calibration Toolbox for Matlab". `vision.caltech.edu/bouguetj/calib_doc/index.html`. Accessed 27 July 2020.

Nistér, D., 2004. "An efficient solution to the five-point relative pose problem". *IEEE transactions on pattern analysis and machine intelligence*, Vol. 26, No. 6, pp. 756–770.

OpenCV, 2021. “Opencv camera calibration”. `docs.opencv.org/3.4.14/`. Accessed 05 June 2021.

PRABURAJ, L., 2018. “Role of agriculture in the economic development of a country”. Vol. 6, pp. 1–5.

ROBOTICS ONLINE MARKETING TEAM, 2019. “Agricultural Robots: Understanding Professional Service Robots in Agriculture”. `robotics.org/blog-article.cfm/Agricultural-Robots-Understanding-Professional-Service-Robots-in-Agriculture/151`. Accessed 13 June 2021.

Rosten, E. and Drummond, T., 2006. “Machine learning for high-speed corner detection”. In *European conference on computer vision*. Springer, pp. 430–443.

Scaramuzza, D. and Fraundorfer, F., 2011. “Visual odometry [tutorial]”. *IEEE robotics & automation magazine*, Vol. 18, No. 4, pp. 80–92.

Shi, J. *et al.*, 1994. “Good features to track”. In *1994 Proceedings of IEEE conference on computer vision and pattern recognition*. IEEE, pp. 593–600.

Trucco, E. and Verri, A., 1998. *Introductory techniques for 3-D computer vision*, Vol. 201. Prentice Hall Englewood Cliffs.

ZAMAN, S., Comba, L., Biglia, A., Aimonino, D.R., Barge, P. and Gay, P., 2019. “Cost-effective visual odometry system for vehicle motion control in agricultural environments”. *Computers and Electronics in Agriculture*, Vol. 162, pp. 82–94.

Zisserman, R.H.A., 2004. “Multiple view geometry in computer vision”.

9. RESPONSIBILITY NOTICE

The authors are solely responsible for the printed material included in this paper.