<http://www.diva-portal.org>

Postprint

This is the accepted version of a paper published in *Biosystems Engineering*. This paper has been peer-reviewed but does not include the final publisher proof-corrections or journal pagination.

Citation for the original published paper (version of record):

Ericson, S K., Åstrand, B. (2018)

Analysis of two visual odometry systems for use in an agricultural field environment.

Biosystems Engineering, 166: 116-125

<https://doi.org/10.1016/j.biosystemseng.2017.11.009>

Access to the published version may require subscription.

N.B. When citing this work, cite the original published paper.

Permanent link to this version:

<http://urn.kb.se/resolve?urn=urn:nbn:se:hh:diva-35853>

Analysis of two visual odometry systems for use in an agricultural field environment.

Stefan Ericson^{a,*}, Björn Åstrand^b

^a*School of Engineering Science, University of Skövde, Skövde, Sweden*

^b*School of Information Science, Computer and Electrical Engineering, Halmstad University, Halmstad, Sweden*

Abstract

This paper analyses two visual odometry systems for use in an agricultural field environment. The impact of various design parameters and camera setups are evaluated in a simulation environment. Four real field experiments were conducted using a mobile robot operating in an agricultural field. The robot was controlled to travel in a regular back-and-forth pattern with headland turns. The experimental runs were 1.8–3.1 km long and consisted of 32–63,000 frames. The results indicate that a camera angle of 75° gives the best results with the least error. An increased camera resolution only improves the result slightly. The algorithm must be able to reduce error accumulation by adapting the frame rate to minimise error. The results also illustrate the difficulties of estimating roll and pitch using a downward-facing camera. The best results for full 6-DOF position estimation were obtained on a 1.8-km run using 6680 frames captured from the forward-facing cameras. The translation error (x, y, z) is 3.76% and the rotational error (i.e., roll, pitch, and yaw) is 0.0482 deg m⁻¹. The main contributions of this paper are an analysis of design option impacts on visual odometry results and a comparison of two state-of-the-art visual odometry algorithms, applied to agricultural field data.

Keywords: Visual odometry, Agricultural field robots, Visual navigation

Nomenclature

α	Field of view of the camera
$\mathbf{R}$	Rotation matrix
$\mathbf{T}$	Relative translation

*Corresponding author. School of Engineering Science. Box 428, 54128 Skövde, Sweden.
Tel: +46 500 448509

Email address: stefan.ericson@his.se (Stefan Ericson)

ω_x	Angular velocity along x-axis
ω_y	Angular velocity along y-axis
ω_z	Angular velocity along z-axis
b	Camera baseline
C_k	Cumulative pose at frame k
d	Disparity between cameras
dTr	Relative pose change of the camera between consecutive frames
f	focal length of the camera
f_s	Frame rate (Hz)
h	Camera height above the ground (m)
k	frame index
T_x	Translational velocity along x-axis
T_y	Translational velocity along y-axis
T_z	Translational velocity along z-axis
v_x	Projected 2D flow field along x-axis
v_x	Projected 2D flow field along x-axis
v_y	Projected 2D flow field along y-axis
v_{max}	Maximum camera velocity (m/s)
x	image points x-coordinate
x_l	x-coordinate of point in left image
x_r	x-coordinate of point in right image
y	image points y-coordinate
Z	Distance to point (depth)
DOF	Degrees of freedom
DW	Downward facing camera
FW	Forward facing camera
ICP	Iterative closest point
IMU	Inertial measurement unit
RANSAC	Random sample consensus
RTK-GPS	Real-time kinematic Global Positioning System
VO	Visual odometry

1. Introduction

Visual odometry (VO) is a method for estimating the position of a camera from an image sequence. In VO, consecutive image frames in a sequence are matched for correspondence and the relative poses between the frames are accumulated. This estimates the travelled path with up to six degrees of freedom (DOF). This technique is applied to agricultural field robots to increase navigation precision compared with that of current GPS navigation systems and to make robots that can operate closer to crops than can current systems.

VO has been around for many years, one of the pioneering studies being by Nistér et al. (2004). Their work introduces an algorithm in which feature points are extracted from the images, matched to each other, and finally used for motion estimation. Outliers among the points are removed using random sample consensus (RANSAC) (Nistér, 2005). The algorithm is applied to a dataset acquired using a vehicle with a forward-facing stereo camera. Many state-of-the-art methods still use the same approach (Kitt et al., 2010; Geiger et al., 2011; Cvišić & Petrović, 2015) but differ slightly in how the features are extracted and matched and in how the outliers are removed. Position estimation can also be improved by making assumptions as to the environment or by adding a motion model of the vehicle, the goal being to reduce the cumulative error.

One reportedly successful method using downward-facing cameras in an agricultural application has been presented by Jiang et al. (2014). Their robot, Gantry, is in the form of a three-meter-high square-shaped table with a combined driving and steering wheel at each leg. Two downward-facing cameras mounted at the top of the robot are used for the VO. In one experiment conducted in a soybean field, the path follows a regular back-and-forth track with a total of 13 headland turns. This experiment has a track length of 2.5 km and consists of 11,700 frames. Their results indicate that the translation error (2-DOF) was under 5.12 m, which corresponds to 0.2% of the travelled distance. A shorter path (i.e., 386 m and 1,300 frames) on a grass road was also evaluated and the reported result is 1.6%.

For urban environments there are publicly available datasets for VO evaluation (Geiger et al., 2012). One such dataset, the KITTI Vision Benchmark Suite, consists of several sequences of images captured from forward-facing cameras mounted on a car roof. Ground truths are available for some, but not all, sequences; the remaining sequences are used for the evaluation of algorithms and a ranking is published online¹. Similar datasets are unavailable for the agricultural case, so a similar comparative ranking does not exist.

Two state-of-the-art methods are selected for evaluation in this paper, the method used with the Gantry robot (Jiang et al., 2014) and the C++ VO library Libviso (Geiger et al., 2011; Geiger, 2015). Gantry was specifically developed for use in agricultural fields and its reported error was under 0.2%. The Libviso

¹The KITTI Vision Benchmark Suite, http://www.cvlibs.net/datasets/KITTI/eval_odometry.php; accessed 20 June 2016.

method is also intended for use in the agriculture field environment (Markt & Technik, 2015). The highest-ranked VO algorithm on the KITTI benchmark (Cvišić & Petrović, 2015) is based on the Libviso method. Cvišić & Petrović
45 (2015) use the same feature extractor but apply a more sophisticated outlier rejector, so only selected feature points are used in the motion estimator; they report translation error of 0.88% and rotational error of 0.0022 deg m⁻¹. The KITTI benchmark list reports 2.44% and 0.0114 deg m⁻¹ translation and rotational errors, respectively, for the Libviso method with stereo cameras, as used
50 in this paper.

To improve the accuracy of VO, this paper seeks new knowledge of cumulative error when VO is used in an open field environment typical for agricultural fields with low-height crops. The accuracy is evaluated by comparing two algorithms, the Gantry and Libviso methods, on both simulated data and on real
55 data captured by a mobile robot. Using both simulated and real field data allows the impact of different design choices to be evaluated, improving our understanding of how various parameters and settings affect the VO results in an agricultural field environment.

The main contributions of this paper are an analysis of design option impacts on visual odometry results and a comparison of two state-of-the-art visual
60 odometry algorithms, applied to agricultural field data.

2. Visual odometry theory

This section presents the theory related to VO. Consider a 6-DOF VO system in which the relative pose change of the camera between consecutive frames is
65 modelled as a rigid motion transform. The transform can be written as shown in Equation 1 (Scaramuzza & Fraundorfer, 2011):

$$dT_r = \begin{bmatrix} \mathbf{R} & \mathbf{T} \\ \mathbf{0} & 1 \end{bmatrix} \quad (1)$$

where $\mathbf{R} \in SO(3)$ is the rotation matrix and $\mathbf{T} \in \mathbb{R}^{3 \times 1}$ is the relative translation. The cumulative pose C at frame k can be obtained using Equation 2:

$$C_k = C_{k-1} dT_r \quad (2)$$

These are the basic equations of the odometry, and the goal is to find dT_r
70 and C_k for each frame. It should be mentioned that the transformation matrix is an overdetermined system and can be rewritten using Euler angles as $\mathbf{v} = (x, y, z, roll, pitch, yaw)$.

The two evaluated methods both use a feature-based approach. Only stereo
75 monochrome perspective cameras are considered, which simplifies the calculation of the distance to the points. Equation 3 shows how the disparities between corresponding points in the left and right images are connected to the distance of the points:

$$Z = f \frac{b}{d} \quad (3)$$

where Z is distance to the point, f is the focal length of the cameras, b is the
80 baseline between cameras and $d = x_l - x_r$ is the stereo disparity between the left
and right images. Furthermore, the images must be corrected for lens distortion
and rectified so that the epipolar lines are parallel to the image x -axis, i.e., both
intrinsic and extrinsic camera parameters are known.

One design parameter is whether a forward- or downward-facing camera
85 should be used. The setting of this parameter is analysed using a theory usually
associated with optical flow (Trucco & Verri, 1998). A camera that moves in a
static environment creates a motion field representing the camera ego-motion.
Let $\mathbf{T} = (T_x, T_y, T_z)$ denote the translational velocity relative to the static 3D
environment and $\mathbf{\Omega} = (\omega_x, \omega_y, \omega_z)$ denote the angular velocity of the camera.
90 The relationship between the projected 2D flow field at image point (x, y) and
the camera's 3D motion can be expressed as shown in Equation 4-5:

$$v_x = \underbrace{\frac{T_z x - T_x f}{Z}}_{\text{Translation}} - \underbrace{\omega_y f + \omega_z y + \frac{\omega_x x y}{f} - \frac{\omega_y x^2}{f}}_{\text{Rotation}} \quad (4)$$

$$v_y = \underbrace{\frac{T_z y - T_y f}{Z}}_{\text{Translation}} + \underbrace{\omega_x f - \omega_z x - \frac{\omega_y x y}{f} + \frac{\omega_x y^2}{f}}_{\text{Rotation}} \quad (5)$$

where (v_x, v_y) are the flow field, Z defines the distance to the point and f is
the focal length of the camera. Note that these equations can be divided into
two parts, the first depending on the camera translation and the second on the
95 camera rotation.

Equation 3 shows that depth Z is inversely proportional to the disparity.
The error of the image point position can be modelled as Gaussian (Matthies
& Shafer, 1987), so the Z error will be large for points far from the camera. At
the same time, Equations 4 and 5 show that points far from the camera, i.e.,
100 $Z \rightarrow \infty$, do not generate any flow in the translational part of the flow field, since
the first term can be neglected. Large flow fields for distant points are therefore
likely to result from rotation of the camera. In contrast, points near the camera
will contain both translation and rotation. These points require better depth
accuracy in order to provide good ego-motion estimation.

105 Consider a vehicle driving forward across an open field. If the cameras are
mounted parallel to the ground, all points will be a similar distance from the
camera. All points can be considered near to the cameras and the error of the
3D points will be fairly small. The 3D points will be much wider ranging if
the cameras are mounted facing forward, including both points near the vehicle
110 and distant points at the horizon. Figure 1 shows the depth distribution of
one million matched 3D points from one of the real field tests. The images
are similar to the two sample images shown in Figure 3. The best theoretical
solution for selecting the camera angle is not obvious, because it depends greatly
on the error of the 3D points. This error in turn depends on the field structure
115 and the algorithm's ability to detect robust points.

Figure 1: Distribution of distance to points: histogram of 10^6 points; maximum distance is limited to 10^6 m.

Figure 2: Maximum overlap for downward-facing camera

One requirement of VO is that consecutive images overlap so that feature points in one image can be found in the next. This is usually not a problem with forward-facing cameras because the focus of expansion is usually visible in the image. For a downward-facing camera, however, this is highly dependent on the maximum velocity. Equation 6 shows the relationship between the maximum velocity and overlap derived from Figure 2:

$$overlap = 1 - \frac{v_{max}}{2hf_s \tan(\frac{\alpha}{2})} \quad (6)$$

where v_{max} is the maximum velocity ($m \cdot s^{-1}$), h is the camera height above the ground (m), f_s is the frame rate (Hz), and α is the field of view of the camera.

3. Difference between algorithms

This section describes the selected VO algorithms in more detail and highlights the differences between them. A comprehensive tutorial on the steps

in the VO pipeline has previously been published in two-parts (Scaramuzza & Fraundorfer, 2011; Fraundorfer & Scaramuzza, 2012) and the methods will be compared according to these steps, as shown in Table 1. In the interest of readability, the methods are described only briefly here; for greater detail, the reader is referred to the original publications (Geiger et al., 2011; Jiang et al., 2014).

Table 1: Evaluated algorithms compared according to visual odometry pipeline steps.

Method	Gantry	Libviso
Image sequence	Stereo DW	Stereo FW
Features	Harris	Blob & corners
Matching	Matching and tracking	Quad matching
Motion estimation	Iterative closest point	RANSAC
	Smooth motion filter	3D-point weight
Local optimisation	Pose graph	None

3.1. Image sequence

The algorithm input is an image sequence from calibrated cameras onboard the robot. This step involves design parameters such as frame rate, resolution, camera type and camera placement. Most real-time algorithms work with a resolution of up to 640×480 , while the frame rate is highly dependent on camera placement and maximum speed (see Equation 6). The Gantry method has only been demonstrated on cameras facing downward, and Libviso has been demonstrated on forward-facing cameras.

3.2. Feature detection

The second step in the VO pipeline is to extract features. It relies on finding salient points that can be described with unique descriptors that are matched between the images. Harris corners are used in the Gantry method (Harris & Stephens, 1988), where they are matched to each other using the correlation of an 11×11 -pixel neighbourhood. Points near each other are suppressed if the corner response is not the maximum in a 20×20 -pixel neighbourhood. That limits the maximum number of feature points on a 640×480 -pixel image to 3072. Libviso, on the other hand, detects both blobs and corners. The images are filtered with 5×5 -pixel templates and both minimum and maximum responses are selected as feature points. Finally, Libviso builds a descriptor at two scale levels based on a Sobel-filtered image. This method generates a large number of matches, i.e., up to approximately 5000 for a well-textured 640×480 -pixel image. This number is then reduced in ensuing matching and outlier removal steps, typically to a few hundred.

3.3. Feature matching/tracking

The Gantry approach uses both matching and tracking. First, features are matched as described previously to find stereo correspondence, which gives a

3D point cloud according to Equation 3. The points in the next frame are then tracked using the Lucas Kanade feature tracker (Bouguet, 2000), which
160 calculates a new point cloud.

In the Libviso method, the descriptors are matched using a circular check for correspondence between all four images in two consecutive image pairs. Outliers are removed in two steps: first, by Delaunay triangulation, which removes points near each other; second, by bucketing, which is a technique for keeping only a
165 certain number of points in each area, resulting in a more even distribution of points over the image (Kitt et al., 2010).

3.4. Motion estimation

The next step in the VO pipeline, motion estimation, makes use of points corresponding to each other in consecutive frames. Even correctly matched
170 points might still belong to other moving objects and not represent camera motion; such points are treated as outliers by both algorithms under consideration. Both evaluated methods differ in how the motion is estimated and in how these new outliers are detected and removed.

The Gantry method estimates motion by linking 3D point clouds to each other using the iterative closest point (ICP) algorithm, i.e., a 3D-3D method
175 according to the VO pipeline. ICP can provide a deterministic solution to the motion estimation problem, instead of the stochastic solution that RANSAC provides. On the other hand, RANSAC can handle more outliers than the ICP algorithm can. The Gantry method tries to solve this problem by removing
180 some of the outliers through the addition of motion constraints to the points. A smooth motion filter removes all points not conforming to the expected motion of the robot. A weighting for the 3D points describing how much each point pair diverges from the expected motion is used to suppress the impact of unexpected motion. Evaluations of various ICP algorithms are presented by Pomerleau
185 et al. (2013). The present work uses the implementation presented by Geiger et al. (2012), which is based on Arun et al. (1987).

Libviso uses a more standard RANSAC approach that minimises the reprojection error of 3D points, i.e., a 3D-2D method according to the VO pipeline. RANSAC is an iterative method in which three 3D points are randomly selected
190 from the previous image pair. The relative motion, dTr , is solved for using the Levenberg-Marquardt algorithm (Marquardt, 1963) so that the projection error is minimal. The motion is then applied to all points to determine the number of inliers, i.e., points with reprojection error below a threshold.

3.5. Local optimisation

The final step in the VO pipeline is local optimisation, which is a technique
195 for minimising the error over several consecutive frames. The Libviso method includes no local optimisation. The Gantry method uses pose graph optimisation (Fraundorfer & Scaramuzza, 2012) with five frames. The motions between multiple frames are calculated according to a scheme and the size of the error
200 determines whether estimation over several frames or cumulative estimation of

consecutive frames should be used. Using five frames for optimisation requires three more pose calculations, which reduces the system’s real-time performance.

4. Materials and method

The experiments were evaluated on two types of datasets, one comprising real field data and one comprising synthetic images generated by a simulator. The real field datasets come from our own experiments representing the agricultural scenario. The design parameters of the experiments are impractical to evaluate through in a real environment and so we turned to simulation to find how the parameters could be adjusted to improve results. Another advantage of using simulated data is that it does provide a very accurate ground truth.

4.1. Mobile robot

A mobile robot based on an electrical wheelchair was used to acquire data on an agricultural field (see Figure 4). The robot has a total of four cameras mounted in two stereo pairs, one pair facing forward and the other facing downward; there is also an omnidirectional camera at the top that was not used in these experiments. Next to the omnidirectional camera is a real-time kinematic GPS (RTK-GPS) and an inertial measurement unit (IMU) used for determining ground truth. Additional calibration runs were conducted to calibrate the sensors’ positions on the robot.

The cameras used are mvBlueFox-120a (Matrix Vision, Oppenweiler, Germany) configured to capture 640×480 -pixel grey-scale images at 20 Hz. The stereo baseline is 200 mm for the forward-facing and 50 mm for the downward-facing cameras (same as used in Gantry). The forward-facing cameras are tilted 12 degrees towards the ground to minimise the amount of sky in the images and to increase the amount of trackable ground. The downward-facing cameras are tilted slightly backwards (98 degrees) to exclude the robot from the cameras’ field of view. Sun covers are used to minimise lens flare, which appears when driving towards the sun. The downward-facing cameras are mounted approximately 1 m behind the forward-facing cameras at a height of 1.1 m. Only a small part of the robot is visible at the top of the image, but shadows from the robot enter the ground cameras’ field of view when driving towards the sun. This will reduce the number of feature points found, as the contrast in the shaded region may be too low.

The RTK-GPS is mounted slightly behind the forward-facing cameras on the top of the robot, together with the IMU. The RTK-GPS is based on a LEA-4T GPS receiver (u-blox, Thalwil, Switzerland) and a high-performance RTK antenna. The system consists of a base station and a rover unit, both using libRTK (Takasu & Yasuda, 2009; Takasu, 2013). These are connected using mobile Internet and the base station is placed within 1 km of the field. A 9-DOF Razor IMU (sparkfun, Niwot, CO, USA) is used to give the orientation of the robot.

All sensors are hardware synchronized. The RTK-GPS runs at 10 Hz while all other sensors are sampled at 20 Hz. Data is collected from all onboard sensors

Figure 3: Sample images from stereo cameras: left- and right-hand images are placed side by side; resolution is 640×480 pixels.

Figure 4: Experimental platform with sensor placement: FW-cam is the forward-facing stereo camera and DW-cam is the downward-facing stereo camera. The omnidirectional camera was not used in the research reported here.

and stored on disks for offline analysis. An additional GPS unit is attached to
 245 provide time measurements of the synchronization pulses, providing valid time
 stamps for all measurements.

4.2. Real field dataset

The field tests were performed on two different fields, one fodder field and
 one oat field. The fodder field consisted mainly of clover, grass, and dandelions
 250 and the average plant height was estimated to be 200 mm. The oat plants in
 the oat field were at various growth stages, so they ranged in height from 20
 to 500 mm. One part of the field was waterlogged during regular sowing, so
 this part had recently been resown. The mobile robot was controlled manually
 during the experiment, traversing the field in a regular back-and-forth pattern.
 255 Figure 5 shows the test fields, the left-hand field being the oat field and the
 right-hand field the fodder field. The figure also shows the surroundings and
 example tracks from two of the experiments taken from the GPS receiver. Each
 field was approximately 300×200 m. The experiments are summarized in Table
 2. Figure 3 shows sample images from both the forward-facing and downward-
 260 facing cameras.

Figure 5: The test field showing GPS tracks of two test runs on Google Earth. Map data: Google, Lantmäteriet, Metria. The left-hand track represents the Test 1 experiment and the right-hand track the Test 3 experiment.

Table 2: Datasets from the agricultural field.

Test label	Field type	Weather	Distance (m)	Frames	Max. speed (mm frame ⁻¹)
Test 1	Oat	rainy	2093	32700	109
Test 2	Fodder	sunny	1838	33400	84
Test 3	Fodder	partly cloudy	2366	58380	66
Test 4	Fodder	partly cloudy	3065	63000	78

The test run on the oat field was conducted during light rain in daytime. Despite using covers to protect the lenses, one raindrop hit the left lens of the forward-facing camera, staying there for half the test. In addition to the raindrop, other real objects appeared, such as a car running along the nearby road and birds taking off when the robot approached. One experiment on the fodder field was performed during the evening when the sun was declining towards the horizon. The low sun was visible to the forward-facing camera, creating lens flares. The sun also created long shadows behind the robot, which are visible in the images from the downward-facing camera. These conditions were not ideal for VO, but are highly representative of real agricultural settings. The other two experiments on the fodder field were conducted during daytime on a partly cloudy day in October. During these experiments, shadows were visible from the forward-facing camera during the headland turns, but no shadows were visible from the downward-facing camera.

4.3. Simulator

The simulator was developed using the Unity game engine (Unity Technologies, Bellevue, WA, USA) in which suitable 3D environments can be built. Cameras are positioned as in the actual robot, with one stereo pair facing forward and one facing downward. The environment is designed to be representative of a corresponding real environment, including mountains, trees, and a blue sky with some clouds (see Figure 6). Real map data is used to obtain the right

Figure 6: Sample image from one of the forward-facing cameras in the real experiment and corresponding sample image from the simulated environment.

height map and right placement of objects. Cameras are mounted at a height of 1 m from the ground.

The path of the camera can be set arbitrarily, but it is particularly interesting to follow exactly the same path as in the real experiments. In that case, the simulator is fed with the ground truth from the real experiment. The simulator also makes it possible to drive in a perfectly straight line on a perfect plane to evaluate various parameters such as sideways slippage, pitch, and roll. This could also be done by others to benchmark their setups. The simulator output is a sequence of synthetic images, and Figure 6 shows a sample image.

4.4. Simulated datasets

The simulator scene is created to represent the real experiment and the path being determined from the ground truth of this dataset. It is important that the robot path follows the height structure of the simulated scene. Otherwise there is a risk of violating the overlap requirement, leading to an increased error. The height resolution of the simulated environment is usually more coarse than in the ground truth, so the solution is to adjust the robot position so it always follow the ground surface of the simulation. The first parameter setup corresponds to the settings of the vehicle used when collecting the data. This means that the results of the real and simulated data should correspond to each other to some extent. In the next step, one design option is changed for each dataset so that the impact of doing so can be evaluated.

- Baseline – affects the accuracy of the 3D points. The baselines used in the experiments reported here are 0.20 m for the forward-facing cameras and 0.05 m for the downward-facing cameras.
- Image resolution – affects the number of feature points that can be extracted and their distribution in space. The cameras used in the present research have a resolution of 640×480 pixels.

- Camera angle – affects the number of feature points and the maximum velocity. The camera facing directly forward has an angle of 0 degrees and the camera facing downward has one of 90 degrees.
- Frame rate – affects the maximum velocity. A higher frame rate enables higher velocity, but more frames may lead to more cumulative error per metre. Gantry datasets are sampled at 10 Hz, while a frame rate of 20 Hz is used in our experiments.
- Algorithm selection - allows a comparison of different feature extractor and motion estimator algorithms. In this research we have only allowed the choice between the Gantry and Libviso methods.

To evaluate if the VO error depends more on the distance between each frame than the cumulative error (i.e. distance travelled) two experiments are performed.

The first experiment is to select a subset of the images (resampling), for example every other image. This creates a data set with fewer frames, but still the same distance travelled. The distance between each frame will then be larger than in the original data set. Note that the relation between the frame rate and the velocity determines the distance between each frame. The same set of images will be produced if the vehicle travels slow and with a low frame rate, as a vehicle with high velocity and high frame rate.

The second experiment is to divide the simulated track into 2, 4, 6 and 8 equally sized parts. This gives a data set with the same number of frames as the previous experiment, but with the same distance between each frame as in the original data set. The average performance of all sub-tracks is used to cover the entire path and to make the result comparable between the two experiments.

If the error is related to the distance between each frame, then this will be seen in the first experiment while the error in the second experiment will be unaffected. This because that the distance between each frame is the same as in the original data set. However if the cumulative error in distance travelled over the whole path is the largest contributing factor to the error then the error is equal for either the first or second experiment.

4.5. Error calculations

The error is calculated in the same way as in the KITTI evaluation. The entire track is divided into sub-sequences 100, 200, ..., 800 m long. Each sub-sequence is analysed by calculating an error vector between the VO result and the ground truth at the endpoint of the sub-sequence. The translation error is the length of the error vector and is presented as per cent of travelled distance. This measure therefore includes errors in the x , y , and z directions. The rotational error includes errors in terms of roll, pitch, and yaw and is calculated as the angle between the VO heading and the ground truth heading. This error is also divided by the travelled distance, giving a measure in deg m^{-1} .

The total error is calculated by averaging all possible sub-sequences that can be extracted from the track. This gives a more representative value of the error

but also requires a more accurate ground truth. Another common approach is just to evaluate the measure at the end point. This gives a more stochastic value as it depends on the travelled path. It can also suffer from error cancellation, giving inaccurate results. This is effect known from wheel odometry calibration (Borenstein & Feng, 1995).

5. Results and discussion

5.1. Agricultural field experiments

The result of applying the different algorithms on one of the real field experiments (Test run 2) is shown in Table 3. The Libviso method is applied on both the forward-facing camera and downward-facing camera. The Gantry method was not successful when applied on the forward-facing camera, due to poorly rejecting outliers.

In total, four test runs were conducted, each several km long. There are high numbers of frames in these tests, which makes the cumulative error dominant. The multiframe optimisation used by the Gantry algorithm is turned off in this test so the results of the two methods can be compared. It has already been demonstrated that multiframe optimisation can improve performance (Jiang et al., 2014).

Table 3: Results of field test on a 1.8 km track on a fodder field (Test run 2). The data set consists of 33400 frames.

VO Algorithm	Camera setup	Translation error	Rotation error (deg m ⁻¹)	Matches (average)
Libviso	FW	8.69%	0.1579	199
Libviso	DW	22.8%	0.3725	217
Gantry	DW	21.0%	0.3716	368

The results for the other test runs showed similar levels of errors. The main error source is poor estimation of pitch and roll. In combination with long accumulation these errors grow until the results are unusable. Still, it can be seen that the forward-facing camera performs much better than does the downward-facing camera, particularly in terms of rotation. The results with the downward-facing cameras are similar between the methods. The pitch and roll are very difficult to estimate and the algorithms have difficulties separating pitch from forward motion when driving forward. This is particularly true with the Gantry method, in which the points are tracked and all points used in the estimation are gathered along the bottom edge of the image. The pitch is then estimated using 3D points with fairly large height errors and a narrow distribution in the motion direction. This leads to poor motion estimation with a pitch bias, causing a huge cumulative VO error. Nourani-Vatani & Borges (2011) stated that pitch and roll cannot be estimated using VO with a camera positioned perpendicular to a flat plane; their solution was to incorporate other sensors for full 6-DOF estimation.

The results can easily be improved by selecting a sub-sample of the data-set, i.e., using a lower frame rate (see Table 4 and Figure 7). This can be done as long as the minimum overlap is not violated, and for these experiments it can only be done with the forward-facing cameras. In this evaluation, every 10th frame was analysed for all tests except the Field 2 test, in which every 5th frame was analysed. The results improve significantly and the best results is 3.76% of travelled distance. These results also include some motion outliers, i.e., wrong motion estimates. The worst case is found in the Test 3 experiment, in which the roll is estimated to be 16.3° instead of 1.3° . That single error increases the total translation error from 3.32% to 3.91%, verifying Howard’s (2008) claim as to the great impact of even a single motion estimation error Howard (2008). A motion model may reduce such errors and improve the performance. The Test 1 experiment used a small number of features, only 70 on average. The rainy weather resulted in a very challenging dataset with low-contrast images, leading to larger error in the motion estimation.

Table 4: Results of all test runs with a sub-sample of the data set using Libviso algorithm on forward-facing camera.

Test setup	Distance (m)	Number of frames	Translation error	Rotation error (deg m ⁻¹)	Matches (average)
Test 1	2093	3270	17.1%	0.0404	70
Test 2	1838	6680	3.76%	0.0482	184
Test 3	2366	5838	3.91%	0.0365	174
Test 4	3065	6300	10.8%	0.0898	152

5.2. Agricultural field simulations

These results are from the simulated dataset where the conditions found in the real experiments are reproduced in the simulator. This dataset is evaluated using both the Gantry and Libviso algorithms. The results are shown in Table 5, and indicates that the error is similar both between the methods and between forward- and downward-facing cameras.

Table 5: Results of simulation of a 1.8 km track on a fodder field (Test run 2). The data set consists of 33400 frames.

VO Algorithm	Camera setup	Translation error	Rotation error (deg m ⁻¹)	Matches (average)
Libviso	FW	2.57%	0.0458	227
Libviso	DW	2.81%	0.0522	233
Gantry	DW	2.85%	0.0504	233

The same data is also used for comparing different combinations of the feature extractors and motion estimators used in the Libviso and Gantry methods. The results are shown in Table 6. Both methods provide very similar results.

Figure 7: Resulting 3D plot from using the Libviso algorithm with forward-facing cameras on Field 2, showing impact of different frame rates; open circle = start position, solid circle = end position.

410 The major difference is that outliers are rejected much better using the Lib-
viso method. The Gantry algorithm is heavily reliant on specifying a good
and limited distance range of the feature points, which works best when us-
ing downward-facing cameras. The Gantry feature extractor (i.e., using Harris
corners) provides more accurate features than does the Libviso method, which
415 uses several types of corners and blobs. This is as expected, as Harris corners
are known to be more accurate than blobs (Fraundorfer & Scaramuzza, 2012).
Gantry’s ICP method for estimating motion is more sensitive to outliers than
is the RANSAC method used in Libviso.

Table 6: Comparison of the feature extractors and motion estimators: visual odometry results for the simulated agricultural field; the angle of the downward-facing cameras is 90° .

Test setup		Translation	Rotation	Matches
Feature	Motion	error	error (deg m^{-1})	(average)
Gantry	Gantry	2.85%	0.0504	233
Gantry	Libviso	2.77%	0.0442	417
Libviso	Gantry	5.25%	0.0963	233
Libviso	Libviso	2.81%	0.0522	233

The impacts of the design parameters are evaluated on the simulated dataset,

420 where only one parameter is changed between each simulation. Table 7 shows the results for the simulated dataset with different design options. The result of the real experiment is included for reference.

Table 7: Visual odometry results of using the Libviso algorithm on a simulated data corresponding to Test run 2 evaluating different setups. Results marked with a * uses an average of shorter distances.

Test setup	Translation error	Rotation error (deg m ⁻¹)	Matches (average)
Reference real			
640×480, baseline = 0.20 m, Cam=12°, Frame rate= 20 Hz	8.69%	0.1579	199
Reference simulation			
640×480, baseline = 0.20 m, Cam=12°, Frame rate= 20 Hz	2.57%	0.0458	223
Baseline = 0.4 m	2.79%	0.0465	181
Baseline = 0.8 m	4.23%	0.0330	108
Resolution 1280 × 480	2.15%	0.0354	238
Cam 0°	2.88%	0.0490	209
Cam 30°	1.69%	0.0275	228
Cam 45°	1.16%	0.0184	230
Cam 60°	1.15%	0.0175	222
Cam 75°	0.92%	0.0149	219
Cam 86°	1.44%	0.0196	204
Cam 90°	2.81%	0.0522	232
Frame rate= 10 Hz (16700)	1.30%	0.0232	219
Frame rate= 8.7 Hz (11133)	0.93%	0.0152	213
Frame rate= 5 Hz (8350)	0.73%	0.0114	204
Frame rate= 4 Hz (6680)	0.64%	0.0101	195
Frame rate= 3.3 Hz (5567)	0.54%	0.0089	187
Frame rate= 2.5 Hz (4175)	1.48%	0.0164	171
Frame rate= 20 Hz* (16700)	2.69%	0.0475	223
Frame rate= 20 Hz* (8350)	3.01%	0.0535	223
Frame rate= 20 Hz* (5567)	3.34%	0.0552	223
Frame rate= 20 Hz* (4175)	3.33%	0.0573	223

425 The results show only minor variations in the results for limited increases in either the baseline or increases in the resolution but surprisingly the results get worse when the base line is increased more dramatically. The impact of changing the camera angle has a larger effect on the result, with the best result obtained around 75°, where there is a good mixture between feature points close to the camera and far away, see Equation 4-5.

430 These experiments also confirm that the error can be decreased by optimising the frame rate in relation to velocity. The test using the resampling experiment where the distance between each frame is larger shows that the error is decreased

for frame rates down to 3.3 Hz. Lower frame rates than that makes the algorithm fail in the turns due to violating the minimum required overlap, which gives an increasing error. For the test using the sub-track experiment the error is more
435 or less the same as in the original test. That means that the error depends more on the distance between each frame than the cumulative error.

6. Conclusions

This paper aims to improve our knowledge of the error resulting when visual odometry is applied in an agricultural field environment with low-height crops.
440 In contrast to urban scenes, agricultural scenes are more open and provide different distributions of feature points. The findings indicate that a camera angle of 75° gives the best results with the least error. An increased camera resolution only improves the result slightly. The main error source is the accumulation of many small errors in the motion estimator. The algorithm must be able to
445 reduce this type of error accumulation either by adapting the frame rate to get an optimal distance between each frame, or by incorporating other sensors that can compensate for the drift over time. The algorithm must also detect and handle large errors in the motion estimation to prevent motion outliers. The findings also illustrate the difficulties of estimating roll and pitch with a
450 downward-facing camera, as it produces feature points with a disadvantageous distribution. The forward-facing camera setup offers better heading estimation, which has a great positive impact on the VO results.

The Gantry method was demonstrated to give similar results to the Libviso method, but it is sensitive to outliers and is less robust. The real experiments
455 produced results with large errors in runs in which the total number of frames exceeded 30,000. The best full 6-DOF position estimation results were obtained with a 1.8-km run using a sub-sample of 6680 frames; in this run, the translation error was 3.76% and the rotational error $0.0482 \text{ deg m}^{-1}$.

This knowledge can be used to design efficient visual odometry systems for
460 vehicles operating in agricultural field environments, which can lead to more robust and more accurate positioning of the robots. This would enable robots to perform more challenging operations closer to the crops such as mechanical weed control.

7. Acknowledgements

465 The authors would like to thank Mariestad Municipality for providing access to the agricultural test fields, and Anna Syberfeldt and Richard Senington for their constructive comments and suggestions on this work.

References

- 470 Arun, K. S., Huang, T. S., & Blostein, S. D. (1987). Least-squares fitting of two 3-d point sets. *IEEE Trans. Pattern Anal. Mach. Intell.*, 9,

- 698–700. URL: <http://dx.doi.org/10.1109/TPAMI.1987.4767965>. doi:10.1109/TPAMI.1987.4767965.
- Borenstein, J., & Feng, L. (1995). Umbmark: a benchmark test for measuring odometry errors in mobile robots. In *Proc. SPIE* (pp. 113–124). volume 2591. URL: <http://dx.doi.org/10.1117/12.228968>. doi:10.1117/12.228968.
- 475 Bouguet, J.-Y. (2000). Pyramidal implementation of the lucas kanade feature tracker. *Intel Corporation, Microprocessor Research Labs*, .
- Cvišić, I., & Petrović, I. (2015). Stereo odometry based on careful feature selection and tracking. In *European Conference on Mobile Robots 2015*. doi:10.1109/ECMR.2015.7324219.
- 480 Fraundorfer, F., & Scaramuzza, D. (2012). Visual odometry : Part ii: Matching, robustness, optimization, and applications. *Robotics Automation Magazine, IEEE*, 19, 78 –90. doi:10.1109/MRA.2012.2182810.
- Geiger, A. (2015). Libviso2: C++ library for visual odometry 2. URL: <http://www.cvlibs.net/software/libviso/>.
- 485 Geiger, A., Lenz, P., & Urtasun, R. (2012). Are we ready for autonomous driving? the kitti vision benchmark suite. In *Conference on Computer Vision and Pattern Recognition (CVPR)*. doi:10.1109/CVPR.2012.6248074.
- Geiger, A., Ziegler, J., & Stiller, C. (2011). Stereoscan: Dense 3d reconstruction in real-time. In *IEEE Intelligent Vehicles Symposium*. Baden-Baden, Germany. doi:10.1109/IVS.2011.5940405.
- 490 Harris, C., & Stephens, M. (1988). A combined corner and edge detector. In *Proceedings of the 4th Alvey Vision Conference* (pp. 147–151). doi:10.1.1.231.1604.
- 495 Howard, A. (2008). Real-time stereo visual odometry for autonomous ground vehicles. In *Intelligent Robots and Systems, 2008. IROS 2008. IEEE/RSJ International Conference on* (pp. 3946 –3952). doi:10.1109/IROS.2008.4651147.
- Jiang, D., Yang, L., Li, D., Gao, F., Tian, L., & Li, L. (2014). Development of a 3d ego-motion estimation system for an autonomous agricultural vehicle. *Biosystems Engineering*, 121, 150 – 159. URL: <http://www.sciencedirect.com/science/article/pii/S1537511014000373>. doi:<http://dx.doi.org/10.1016/j.biosystemseng.2014.02.016>.
- 500 Kitt, B., Geiger, A., & Lategahn, H. (2010). Visual odometry based on stereo image sequences with ransac-based outlier rejection scheme. In *IEEE Intelligent Vehicles Symposium*. San Diego, USA. doi:10.1109/IVS.2010.5548123.
- Markt & Technik (2015). Auch in der landwirtschaft schreitet die automatisierung und digitalisierung voran kameras als augen des jät-roboters. In *Markt & Technik* 49 (pp. 55–56). WEKA Fachzeitschriftenverlag GmbH. URL: www.elektroniknet.de.

- 510 Marquardt, D. W. (1963). An algorithm for least-squares estimation of nonlinear parameters. *Journal of the Society for Industrial & Applied Mathematics*, 11, 431–441. doi:10.1137/0111030.
- Matthies, L., & Shafer, S. (1987). Error modeling in stereo navigation. *Robotics and Automation, IEEE Journal of*, 3, 239 –248. doi:10.1109/JRA.1987.1087097.
- 515 Nistér, D. (2005). Preemptive ransac for live structure and motion estimation. *Mach. Vision Appl.*, 16, 321–329. URL: <http://portal.acm.org/citation.cfm?id=1107685.1107693>. doi:10.1007/s00138-005-0006-y.
- Nistér, D., Naroditsky, O., & Bergen, J. (2004). Visual odometry. *Computer Vision and Pattern Recognition, IEEE Computer Society Conference on*, 1, 652–659. doi:<http://doi.ieeecomputersociety.org/10.1109/CVPR.2004.265>.
- Nourani-Vatani, N., & Borges, P. V. K. (2011). Correlation-based visual odometry for ground vehicles. *Journal of Field Robotics*, 28, 742–768. URL: <http://dx.doi.org/10.1002/rob.20407>. doi:10.1002/rob.20407.
- 525 Pomerleau, F., Colas, F., Siegwart, R., & Magnenat, S. (2013). Comparing icp variants on real-world data sets. *Autonomous Robots*, 34, 133–148. URL: <http://dx.doi.org/10.1007/s10514-013-9327-2>. doi:10.1007/s10514-013-9327-2.
- 530 Scaramuzza, D., & Fraundorfer, F. (2011). Visual odometry [tutorial]. *Robotics Automation Magazine, IEEE*, 18, 80 –92. doi:10.1109/MRA.2011.943233.
- Takasu, T. (2013). Rtklib: An open source program package for gnss positioning. URL: <http://www.rtklib.com/>.
- 535 Takasu, T., & Yasuda, A. (2009). Development of the low-cost rtk-gps receiver with an open source program package rtklib. In *International Symposium on GPS/GNSS, International Convention Center Jeju*. Korea. doi:10.1.1.472.7544.
- Trucco, E., & Verri, A. (1998). *Introductory Techniques for 3-D Computer Vision*. Upper Saddle River, NJ, USA: Prentice Hall PTR.

540 List of Figures

1	Distribution of distance to points: histogram of 10^6 points; maximum distance is limited to 10^6 m.	6
2	Maximum overlap for downward-facing camera	6
3	Sample images from stereo cameras: left- and right-hand images are placed side by side; resolution is 640×480 pixels.	10
- 545

4	Experimental platform with sensor placement: FW-cam is the forward-facing stereo camera and DW-cam is the downward-facing stereo camera. The omnidirectional camera was not used in the research reported here.	10
550	5 The test field showing GPS tracks of two test runs on Google Earth. Map data: Google, Lantmäteriet, Metria. The left-hand track represents the Test 1 experiment and the right-hand track the Test 3 experiment.	11
555	6 Sample image from one of the forward-facing cameras in the real experiment and corresponding sample image from the simulated environment.	12
7	Resulting 3D plot from using the Libviso algorithm with forward-facing cameras on Field 2, showing impact of different frame rates; open circle = start position, solid circle = end position.	16

560 List of Tables

1	Evaluated algorithms compared according to visual odometry pipeline steps.	7
2	Datasets from the agricultural field.	11
3	Results of field test on a 1.8 km track on a fodder field (Test run 2). The data set consists of 33400 frames.	14
565	4 Results of all test runs with a sub-sample of the data set using Libviso algorithm on forward-facing camera.	15
5	Results of simulation of a 1.8 km track on a fodder field (Test run 2). The data set consists of 33400 frames.	15
570	6 Comparison of the feature extractors and motion estimators: visual odometry results for the simulated agricultural field; the angle of the downward-facing cameras is 90°.	16
7	Visual odometry results of using the Libviso algorithm on a simulated data corresponding to Test run 2 evaluating different setups.	17
575	Results marked with a * uses an average of shorter distances. . .	