TrimBot2020 Deliverable D2.5

Evaluation Manipulator and Tools V2, Closed-Loop Motion Planning

Principal Author: Wageningen University (WU) & Wageningen Research (WR)
Contributors: A.-Ludwigs-Univ. Freiburg (ALUF)
University of Edinburgh (UEDIN)
Dissemination: PU

Abstract. This deliverable describes the evaluation of the second version of the end-effectors for bush trimming and rose stem clipping, as well as the vision modules and control components related to this. For the bush trimming end-effector the implementation of a flexible shaft for the cutter drive system turned out to be not successful. The cameras of the bush trimmer were calibrated, including an accurate hand-eye calibration method. Images acquired with this setup allowed to perform a multi-viewpoint target shape fitting. The performance of the vision-based open-loop and closed-loop trajectory control of the system has been tested with various shapes of bushes, using artificial and real bushes. The pipeline for automatic scanning, trimming and evaluation of the trimming results works as expected. For stem clipping, a static rose clipping scenario was evaluated with 3D position based visual servoing to red markers on real rose bush stems. A CNN based method was used to segment the bush. The visual servoing trajectory control first scans the bush from multiple viewpoints and then position-based control is used to navigate towards the targets. Finally the stem is cut at the target position. Results of branch segmentation, bud eye detection are presented and discussed. The performance of the visual servoing maneuver and stem clipping action is evaluated.

Deliverable due: Month 36

Contents

1 Introduction

This deliverable describes the evaluation of the 2nd version of the end-effectors for bush trimming and single stem clipping (roses), as well as the vision modules and control components for closed-loop bush trimming and stem clipping, which were previously described in *D2.4 - Manipulator and tools V2, Closed-Loop Motion Planning*. In section 2 the evaluation of the bush trimming system is described. Most elements here related directly to the accompanying paper [6], which is added as a confidential appendix, and therefore are only summarized. In section 3, the evaluation of the rose clipping system is described.

2 Bush trimming evaluation

This section contains a brief overview of the bush trimming evaluation, and relates the current status with the previous deliverable 2.4. More details and results are given in the accompanying paper [6], which is also added as an appendix to this deliverable. A demonstration video clip of topiary bush trimming is available on the Trimbot2020 YoutTube channel [3].

2.1 Material and Methods

2.1.1 Cutter Drive System

In deliverable 2.4, it was indicated that to reduce the weight of the end effector, the drive motor would be placed on the 4th axis. To the drive the cutter head, a flexible drive shaft would be used, which still had to be designed. Since then, this design has been realised and implemented on the robot, as shown in Fig. 1.

While testing the bush cutter using this drive system, several observations were made:

1. The drive shaft is able to drive the cutter head. However, the torque it can provide turned out to be insufficient to properly cut the bush. When trimming a bush the rotations of the cutting blades get blocked.
2. As the drive shaft has limited length and relatively high stiffness, it is strongly affecting the freedom of movement of the arm.
3. For multiview image data collection with the tool mounted camera needed for the shape fitting method, the arm poses must be (manually) chosen such that the shaft is not seen by the cameras.
4. The DeepTAM method that is planned to be used for visual treacking of the arm (see details in Deliverable 2.4) can probably not deal with a dynamic object being (quite prominently) visible; it assumes a static scene.

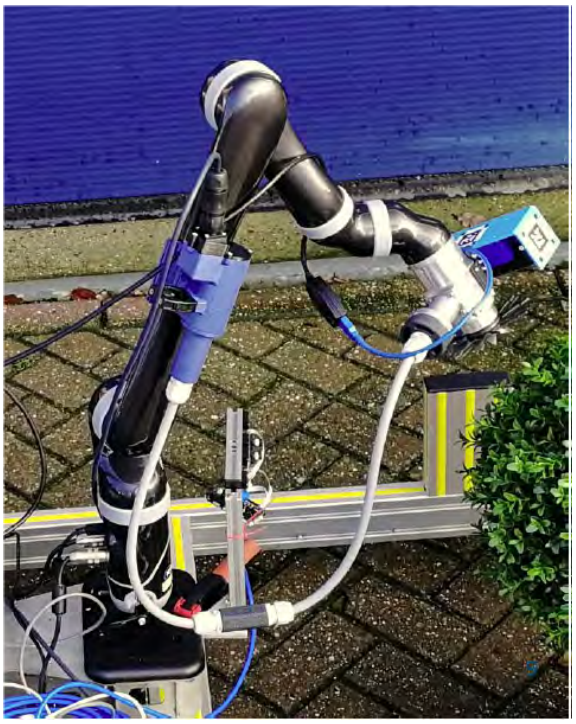


Figure 1: Realized bush trimming drive system using a flexible shaft to drive the cutter tool

Implications of these observations are as follows:

1. This situation might be resolved by adjusting the design, either by selecting a shaft that can handle larger torques or increasing the rotational speed of the drive shaft so a lower torque is required. Both options do require significant design modifications, and are not guaranteed to solve the problem while keeping the expected benefits of this approach.

2. This situation can be reasonably handled by manual control, in which there is direct visual feedback of the system's status to the user, thus avoiding too much tension on the drive shaft or it becoming entangled on the arm. Using automated control is much harder, as the visual feedback is missing and this can only be achieved by limiting the allowed arm configurations. Although driving the cutter head directly also requires an electrical cable to be placed along the arm, this seems more suitable. As this cable can be kept much longer and has larger flexibility, the implications of entangling this somewhat around the arm are much smaller, and are not expected to limit the arm behaviours during a single trimming task.

3. This observation is probably not a significant complication, because currently only specific static poses are used, although care should be taken to plan them carefully. It could also mean that changing the shaft mounting invalidates existing scanning trajectories.

4. Unlike the above point, this situation cannot easily be fixed by trajectory planning as it seems very hard to avoid visibility of the shaft in the wide-angle cameras, and because DeepTAM uses images throughout the entire trajectory and not just at select poses.

As result of this, it was decided to revert the design to the previous stage, in which the cutter head was driven directly by the motor. Using improved drive shaft is still an option for a future generation. however, the implications this approach has for other elements of the system might also result in abandoning this concept.

2.1.2 Camera setup for bush trimming

The camera setup as described in deliverable 2.4 was realized with several minor adaptations:

- The back cover was adapted by moving it further outwards, to allow more space for the flat cables connecting the cameras.
- The design of the back cover was adjusted to accommodate placement of larger April tags on multiple sides of the housing (top, rear and sides).
- The camera was rotated 90 degrees towards the drive system, to allow more free space around the cutting tool.

The resulting camera setup is shown in Fig. 2. A DispNet stereo-disparity estimation network [8, 4] is used to get depth data from the lower camera pair (see Fig.4). The camera setup is rigidly mounted on the trimming tool; the viewing axis is offset to avoid occlusions by the tool blades.

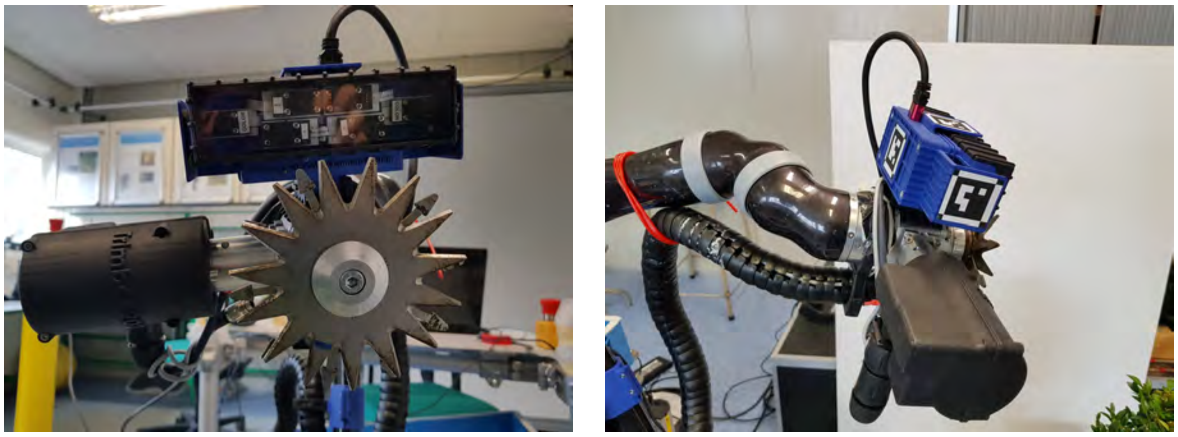


Figure 2: Modified camera setup for bush trimming.

2.1.3 Vision module

The vision module acquires camera images and pose information; it produces a trimming target shape (as triangular mesh) which is then further processed by the arm motion planner. To be able to plan a trimming motion, the location of the target surface with respect to the robot system must be known. While the garden map and robot-localization provide a location prior for both the robot and the target bush, both these locations are expected to be too inaccurate to serve as starting position for the bush trimming operation performed by the robotic arm. It is therefore desirable that the target surface information can be dynamically adjusted to the observed scene. We choose a vision-based approach: we reconstruct a 3D model of the rough location of the target bush in front of the robot. A 3D shape of the desired cutting shape is then fitted into the reconstruction, using a Trimmed-ICP algorithm [1] which is robust to only partial overlap of given shape and target shape (the geometry and size of the target bush are specified by the user). More details on this approach are given in [6]

An input/output example for the vision module is shown in figure 4.

We found that a single viewpoint of the target bush does not reliably yield a good fit, especially with overgrowth. As a remedy, a fixed scanning trajectory for the arm was implemented to cover the scene in front of the robot from multiple viewpoints. The vision module combines information from these viewpoints by accumulating multiple views' depthmaps (via the DispNet) into a single cumulative pointcloud. We found that this makes for reliable fitting. An example of shape-fitting progression is shown in Fig. 5.

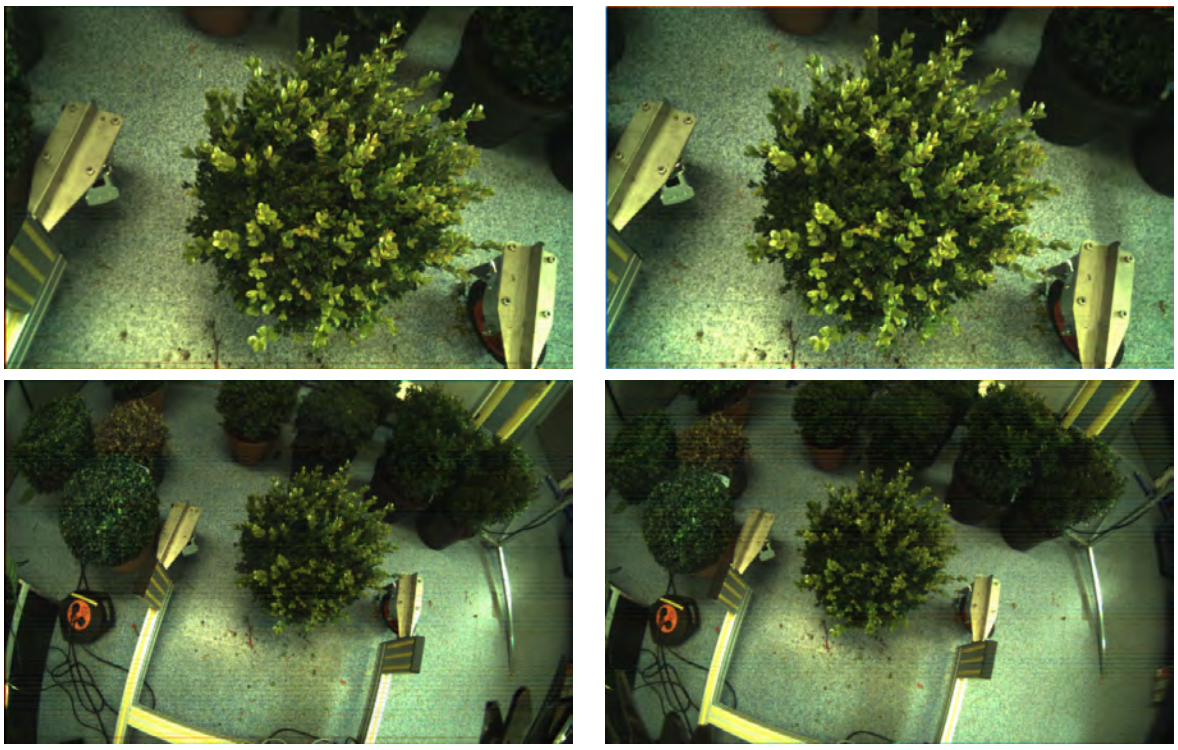


Figure 3: Raw image data from the multi-camera setup: Shown are two standard views (**top row**) and two wide-angle views (**bottom row**).

2.1.4 Hand-eye calibration

As shown in Fig. 2 the camera unit is placed on top of the trimming tool, close to the robot's TCP. In order to control the arm based on the information from the camera the pose of the

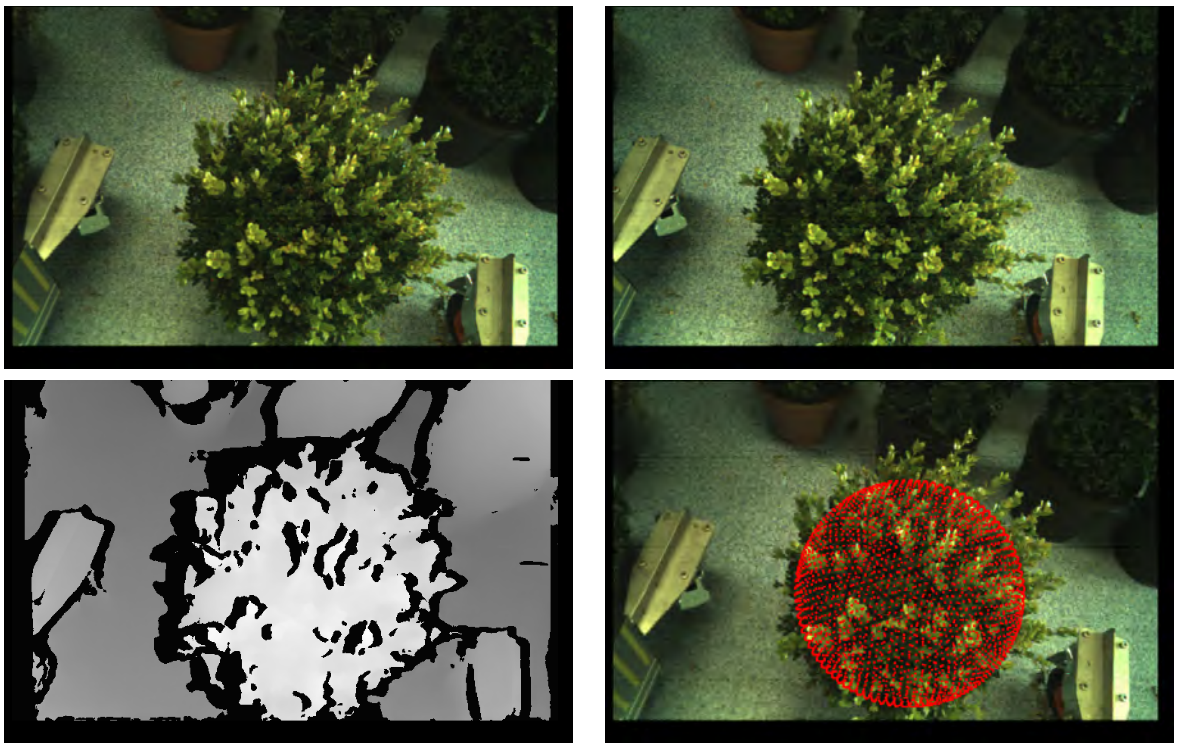


Figure 4: Results of disparity estimation and bush-model fitting: The DispNet takes rectified stereo images (**top row**) and produces disparity maps (**bottom left**) which are used by the shapefitting module to fix a bush location in the observed scene (**bottom right**).

camera with respect to the TCP is needed with high accuracy. To obtain this pose a hand-eye calibration was done using the commercial HALCON software (Version 13, MVTec, Germany). The hand-eye calibration is based on providing multiple images of a known calibration object. In this specific case an original HALCON hexagon calibration plate type 320mm was used. The calibration object is placed on a fixed position and the robot moves the tool mounted camera over the stationary calibration object. The pose of the robot tool for each calibration image is used for the calibration. This results in a chain of coordinate transformations. In this chain, two transformations (poses) are known: the pose of the robot tool in robot base coordinates and the pose of the calibration object in camera coordinates, which is determined from the calibration images. The hand-eye calibration then estimates the other two poses, i.e. the relation between the robot and the camera and between the robot and the calibration object, respectively. Five to ten images of the target were collected using manual arm control, varying the arm pose to maximize the variance in the camera pose relative to the target (calibrations may fail if the recorded poses do not cover all degrees of freedom). If the arm poses are noisy, calibration profits from more data, i.e. more images, for regularization. The software also produces an error measure which helps judge whether the calibration procedure was successful. As reference camera the left camera of the bottom row camera pair was used, as shown in Fig. 6. Fig. 7 shows screenshots from the hand-eye calibration procedure.

Result of the hand-eye calibration is a rotation and translation vector of the camera to the TCP position of the robot, such as the example shown here:

```
# Rotation angles [deg]:
```

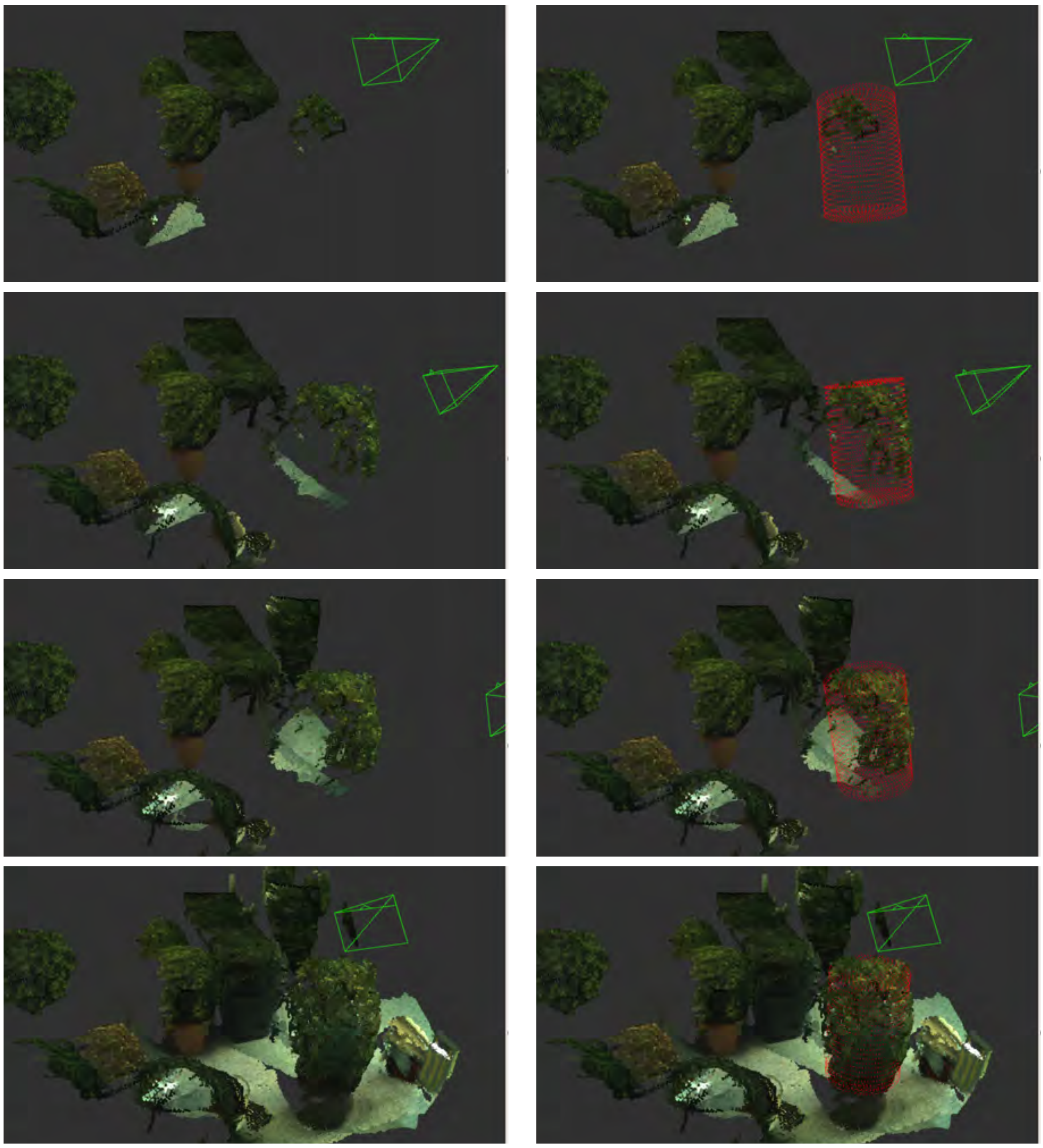


Figure 5: Data collection from multiple viewpoints improves shape fitting: **top-to-bottom:** 1, 2, 3, and 10 views merged into a single pointcloud. Each new viewpoint's data complements the prior pointcloud (**left column**). In this case, a cylinder shape is fitted to a not-really cylindrical plant; for a stable result, many views must be used (**right column**).

```
r 180.73 359.06 269.55

# Translation vector (x y z [m]):
t 0.0235 0.078 -0.0810
```

Intrinsic and stereo-extrinsic calibration was done using Kalibr [7]. In the current implementa-

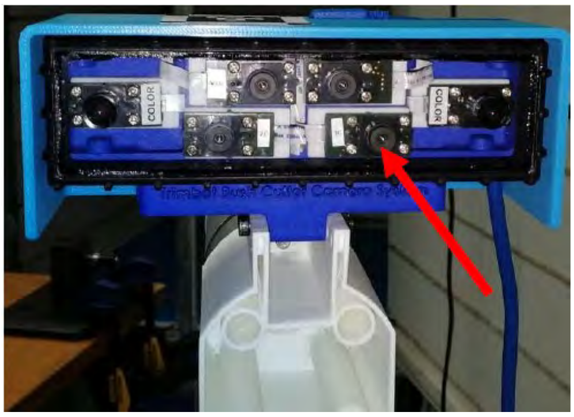


Figure 6: Reference camera used for hand-eye calibration (indicated by arrow).

tion the the hand-eye calibration procedure using HALCON requires many manual steps, such as manually recording robot poses. It is desired to develop a more automatic procedure, as hand-eye calibration will be needed every time after remounting the tool and/or cameras onto the arm.

2.1.5 Arm control issues

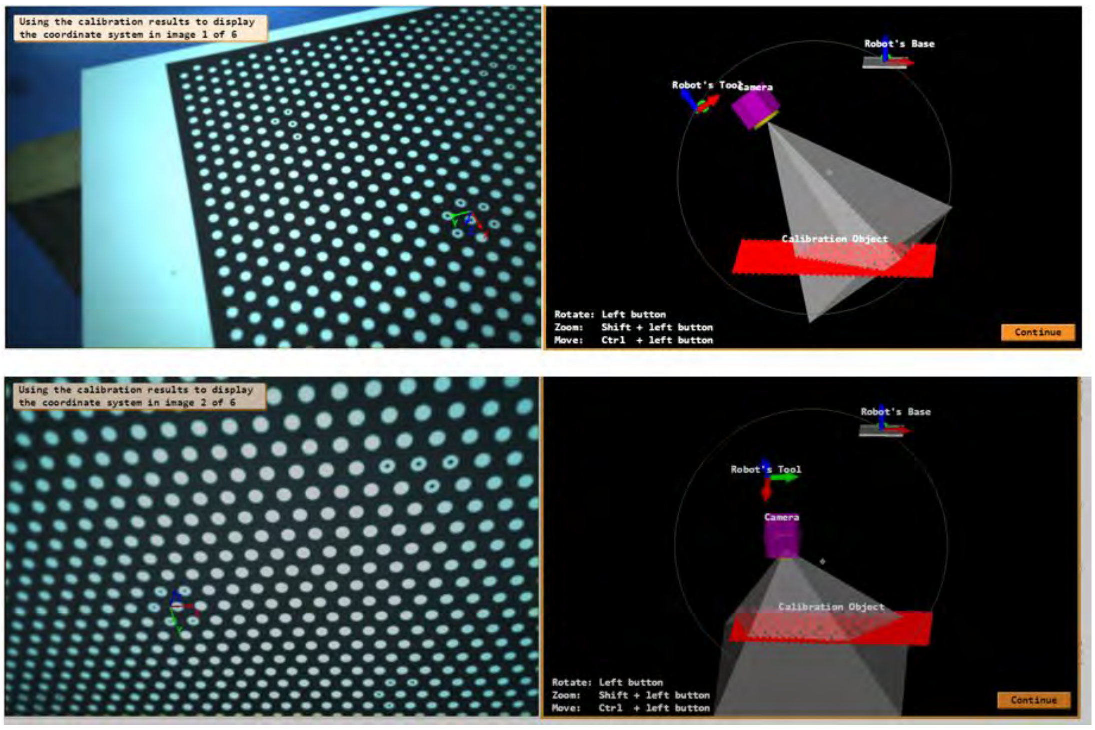
In deliverable 2.4, several issues were mentioned with respect to the accuracy of the Kinova robot arm. Given the improved results from camera- and hand-eye calibration, the observations seem to match much better, and the effect of arm inaccuracies seems to be limited. Thus, the steps for improving control accuracy mentioned in deliverable 2.4 are put on hold, except for the development of the DeepTAM approach, which is still on-going.

2.1.6 Vision-based open-loop trajectory control

For the previous version of the open-loop trimming, the path was planned based on an estimate of the bush that was scanned before and provided to the planning algorithm at the time of execution. For estimating the bush, a single viewpoint approach was used, as described in deliverable 2.2, with the disadvantage that less than 50% of the bush could be properly observed. To overcome this, a multi-viewpoint scanning procedure was introduced, which observes the bush from the front, left and right using viewpoints placed above and around the bush' centre. By using a fixed scanning pattern, and stopping at the positions where a view of the bush should be captured, the information from the cameras can be merged into a point cloud of the bush. Based on this point cloud, a mesh describing the target shape of the bush is fitted, as described in section 2.1.3. more details on this are given in [6].

The fitted mesh is then used to plan a trimming path using the procedure described in Algorithm 1. More details about how such procedure was devised are in [5].

First, a `robotics.InverseKinematics` object is built in Matlab based on the robot URDF, and used to generate inverse kinematic solutions. The Edge Distance Function (EDF) between each pair of candidate configurations is defined as such:


Using the calibration results to display the coordinate system in image 1 of 6

Using the calibration results to display the coordinate system in image 2 of 6

Figure 7: Screenshots from the hand-eye calibration procedure showing camera images of the calibration target (left column) and visualized corresponding pose of robot and camera with respect to calibration target (right column).

1. If the pose distance between the associated bush locations is lower than a threshold Δ_{max}, and the associated movement direction is not downward, EDF was defined as $||QQ_i - QQ_j||$.
2. Elsewhere, EDF is assigned a threshold value. Such value indicates the need of a non-cutting transition between the given bush locations.

In an ideal bush trimming motion, the trimming point of the trimming tool should be the one closest to the bush, with other parts of the tool further away. The movement of the trimming tool would then be such that the trimming point is placed most upfront in the direction of movement, thereby grasping and cutting branches before they come in contact with other parts of the trimming tool. As the current trimming tool could be used for omnidirectional trimming, in these first tests this constraint is relaxed and side cutting motions are allowed as well.

2.1.7 Closed-loop trajectory control

If the previous part is completed successfully, feedback on the trimming results can be added to generate a closed-loop trajectory. For that, the same scanning procedure as used before is executed again. By comparing the results of the second scan with the first scan, it is possible to determine which parts of the bush have been affected by the trimming operation. This is done through pointclouds comparison. If this information is then also compared with the mesh

```
**Input:** Target mesh $M$, Robot model $R$, Tool model $T$
**Output:** Sequence of robot configurations $Q$
    *Initialize bush locations $L \leftarrow \emptyset$, candidate robot configurations $Q_{all} \leftarrow \emptyset$, scheduled robot configurations $Q \leftarrow \emptyset$*
    for face $F$ in $M$ do
        for pose $P$ in $L$ do
            if (patch_distance$(F, P) < \sigma\Delta_{max}$) then
                update $P$ according to $F$
            else
                $L \leftarrow L \cup$ pose$(F)$
    for pose $P$ in $L$ do
        if $(dist(P) > R_{lim})$ then
            $L \leftarrow L \backslash P$
    for pose $P$ in $L$ do
        for $i = 1$ to $MAX\_CONF$ do
            $Q_{all} \leftarrow Q_{all} \cup$ IK$(P, R, T)$
    for every pair $Q_1, Q_2$ in $Q_{all}$ do
        compute EDF$(Q_1, Q_2)$ according to planning cost function
    $BEST\_COST \leftarrow \infty$
    for $i = 1$ to $MAX\_ITERATIONS$ do
        generate $N\_ANTS$ schedules of $T$ with ACS
        for $j = 1$ to $N\_ANTS$ do
            $[Q_j, COST_j] \leftarrow$ shortest_path(graph $j$)
            if $(COST_j < BEST\_COST)$ then
                $BEST\_COST = cost_j$
                $Q = Q_j$
    for $Q_i$ in $Q$ do
        if (EDF$(Q_i, Q_{i+1}) \geq THRESHOLD$) then
            replace transition from $Q_i$ to $Q_{i+1}$ with non-cutting path
```

Algorithm 1: Coverage trajectory planning algorithm.

that indicates the desired shape of the bush, it is possible to determine whether the trimming action was at the correct depth. If the bush was trimmed too shallow, the trimming path can be adjusted by modifying the robot configurations in the plan by exploiting the approximate relation $q_{new} - q \approx J(q)^{\#}(p_{new} - p)$ for small $(p_{new} - p)$, where $J(q)^{\#}$ is the pseudoinverse of the kinematic Jacobian of the robot. $(p_{new} - p)$ is defined as a small constant step towards the target mesh, to make a slightly deeper trimming move. A limit in the number of trimming attempts is needed, to prevent the system from being stuck on the same trimming segment if the cut keeps on being evaluated as too shallow.

If the trimming was too deep, an adjustment of the mesh is necessary to preserve the shape of the bush. Next, also the trimming of the following patch should be executed more shallow, to avoid shrinking the bush too much.

Furthermore, this information can be used for adapting properties of fitting and trimming on the longer term, to avoid that the next bush is also cut too shallow.

2.1.8 Test set of bushes

To test the performance of the integrated open- and closed loop trimming of bushes multiple shapes have been used. Initially, work was mainly done on spherical bushes, but for this stage also other shapes such as cylindrical and cube shaped bushes are used. First tests for both shape fitting and bush trimming used artificial bushes, as these allow better evaluation if the approach is functioning correctly. After this, also real bushes were evaluated. For fitting, also more deviating shapes have been considered, such as a larger cube (combining 4 smaller cubes into a block), a hedge (created from 2 smaller cubes) and the combination of a cube with a sphere.

2.1.9 Evaluation

Evaluation of shape fitting and trimming was done using the data acquired in the scanning steps before and after trimming. For shape fitting evaluation, the 3D points of the scene reconstruction were compared to the fitted ideal shape. By analysing their distribution, a measure of fitting quality could be obtained. For trimming evaluation, the same approach was used, but now comparing the scanning results before and after trimming. For more detail on the evaluation procedure, see [6].

2.2 Results & Discussion

2.2.1 Bush fitting

Using the new camera calibration and scanning approach, the bush fitting results have clearly improved. To quantitatively test the bush fitting module, several bush shapes were used. This included artificial, real pre-trimmed bushes, and outgrown bushes, with spherical, cubical, and cylindrical shapes. Furthermore, some composite shapes were added to test fitting performance on non-symmetrical objects.

A summary of the quantitative results is given here, with more detail and pictures of some of the evaluated bushes in [6]. For qualitative inspection, also other shapes were used, such as a large block and a small hedge (consisting of 4 and 2 cubical bushes). Examples of the bushes used in the evaluation are shown in Fig. 8.

Artificial bushes The ideal case for fitting is a smooth shape that is already close to the target shape. Testing an artificial sphere, pinecone, and cube shows that a single viewpoint is already sufficient to reconstruct the target shape, with the majority of the points within 0.6cm from the fitted mesh or further away than 15cm. Points further away than 15cm were excluded from the fitting procedure as they belong to objects of the surrounding scene and not to the target object. For further details is refered to [6].

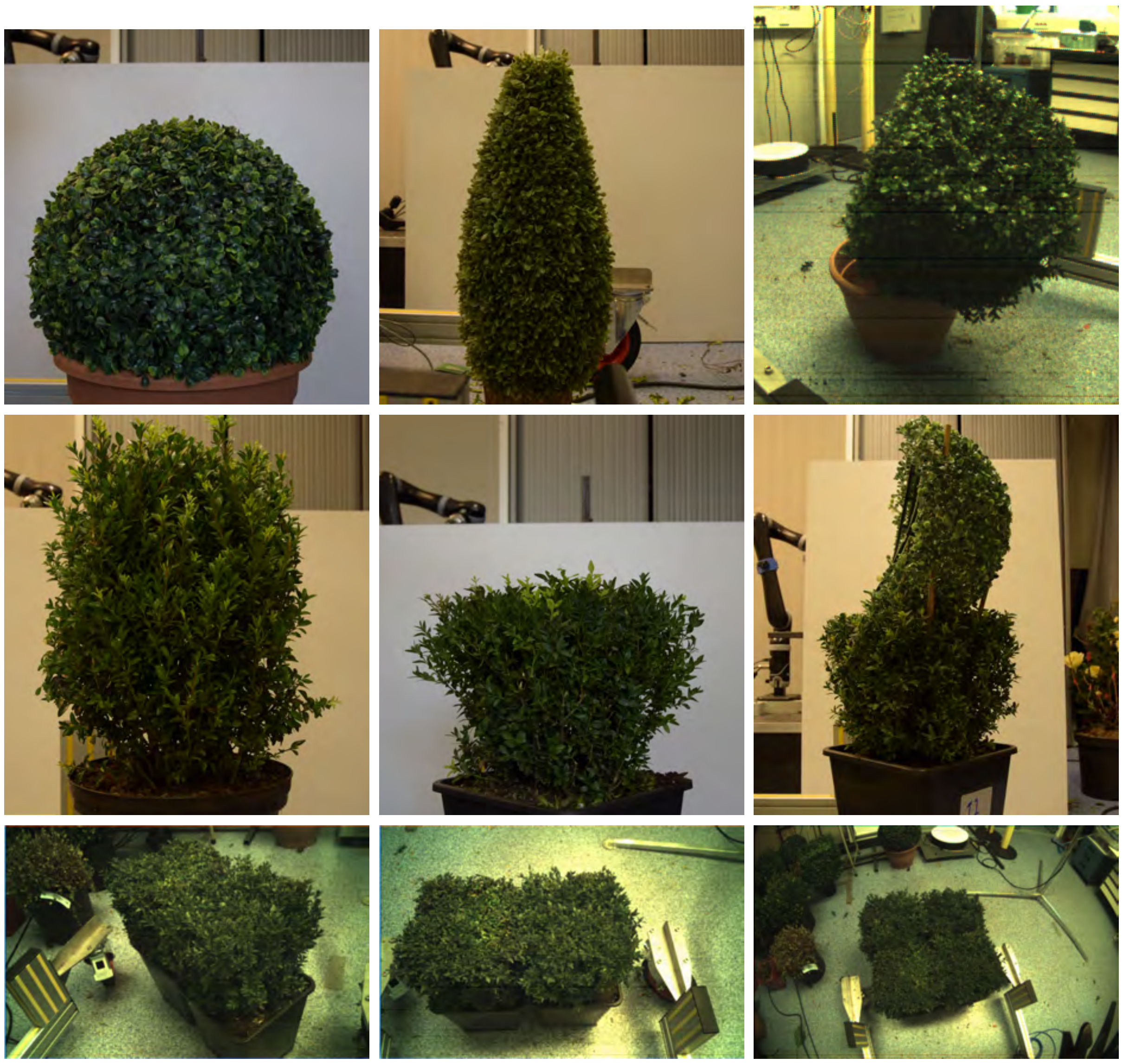


Figure 8: A sample of the bushes used in evaluation. The top row contains artificial bushes (sphere, pinecone and rotated cube). The middle row contains two real bushes (cylinder and cube) and and a composite shape of a real cube and a artificial half-sphere. The bottom row contains two hedges (with different orientation) and large block, all composed from real cubical bushes.

Real plants and complex shapes When using more complex shapes and real plants (which do not perfectly match the target shape), more viewpoints are required to get a good shape fit. For the cylinder, for example, a stable result was only obtained after fusing nine views. In Fig. 9, a cylinder-shaped boxwood bush and an artificial complex shape (made up of a cube and a hemisphere on top) were evaluated. Here, the final pose of the target shape conformed to the true position and orientation of this bush. Similar results were observed with the other shapes tested. Due to noise in depth data and irregularities in the scanned objects fitting accuracy decreased, although most points stay within 5cm of the fitted mesh. Given the higher variation in the shape of the real bushes compared with the shape of the artificial bushes these results do make sense. For further details it is again refered to [6].

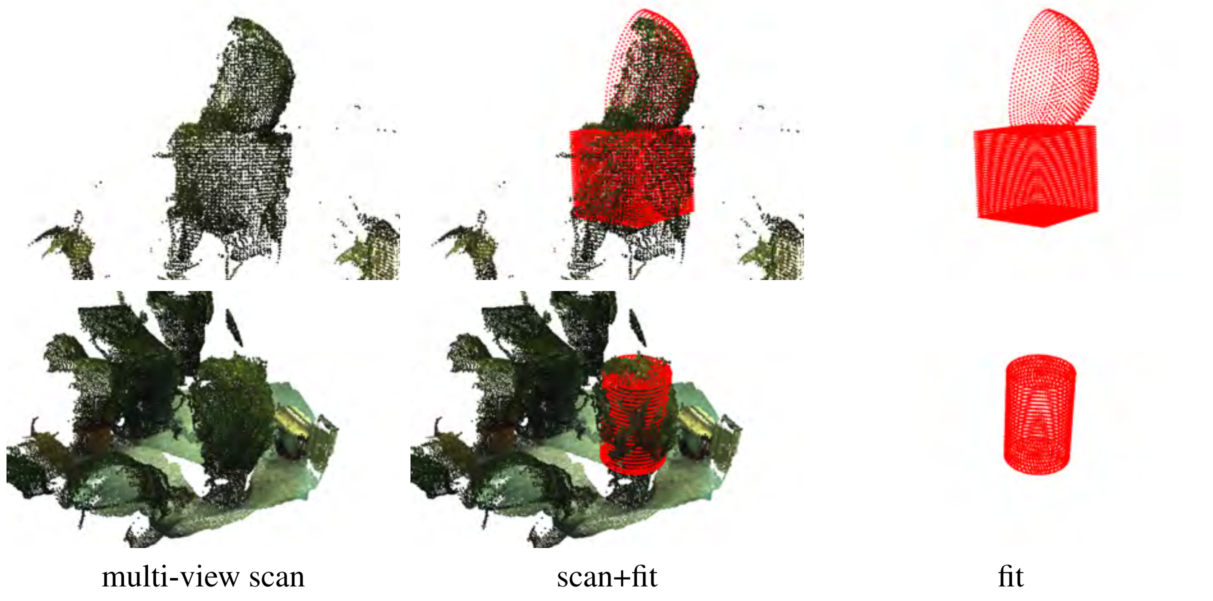


Figure 9: Fitting evaluation of a real cube bush with an artificial hemisphere fixed on top, and a real cylinder-shaped bush. As highlighted in Fig. 5, multiple viewpoints needed to be fused for these shapes to fit well. The cylinder bush did not conform well to its target shape, as it was slightly tilted.

2.2.2 Bush trimming

Based on the fitted mesh, a trimming path was generated using the method described in 2.1.6, and sent to the arm control for execution. The resulting path can be observed in the demonstration video [3], while pictures of the trimming result for a cubical bush are given in Fig. 10. This path consisted of trimming motions in which the trimming tool moved over the bush while trimming a given patch, and connecting motions, in which the arm was moving to and from the bush to go from one trimming patch to the next. During testing of a cubical bush, it was observed that finding suitable trimming motions is easier if the robot base is at the edge of the bush compared to being at the side of a bush.

From visually observing the experiments, it was found that the trimming patches provided a good coverage of the reachable parts of the bush, although not all motions conformed well to the desirable behaviour as described in section 2.1.6. Next to that, especially the connecting motions sometimes showed unpredictable behaviours. Furthermore, some inconsistencies were observed in the trimming paths in case they seemed to be either too close or too far away from the trimming surface. As this is hard to judge while the arm is moving, a more detailed tracking of the arm and tool is desired to evaluate this. Finally, in a number of cases it was observed that branches sticking out of the bush were pushed aside instead of being trimmed. The exact reason for this remains unclear, but it seems that both the approach of these branches by the tool and the design of the end-effector play a role in this. As result, the trimming performance might have been lower than expected, although it still demonstrates the functionality of the concept. Improved trimming performance is expected by better matching the planner behaviour to the desired cutting paths and updating the trimming tool design to avoid branches being pushed aside.

2.2.3 Evaluation

To evaluate trimming performance, the histogram of scene point distances from the fitted mesh before trimming (see for examples [6]) is compared to the corresponding histogram after trimming. Again, a summary of the results is provided here, while a more detailed evaluation of trimming performance is given in [6]. Evaluation was performed using a single scan of the target surface. The difference as result of trimming was then calculated by comparing the scans before and after trimming. If the trimming result was considered insufficient, the same trimming path was repeated up to 3 times, while otherwise the system advanced to the next part of the bush. In a future extension of this method, the observed difference as result of trimming will also be used to adapt the trimming motions such that the missing areas will also be treated properly.

For evaluating trimming quality, the full scanning trajectory was used, such that a more complete view of the bush was obtained. From the resulting histograms, it was observed that the point distances change after trimming, by increasing the percentage of points having a low distance from the target mesh. This indicates that the trimming has an effect of the bush, especially for points being more than 3cm away from the mesh describing the desired shape. This can also be observed from Fig. 10, where most of the extended branches for the cubical bush are removed after trimming. For the sphere, outgrowth was removed on the left and right side of the bush, but some branches remained. Since not all parts of the bush could be reached by the arm, some untrimmed areas of the bush remain present in this data.

2.3 Conclusions

As described above, an integrated pipeline for observing and trimming boxwood bushes was realized and evaluated. From this, the following can be concluded:

- The design of the flexible driveshaft did not work out as expected, and needs further evaluation and possibly re-design. As a remedy the previous design (motor placed close to the tool) was used for the evaluation described here.

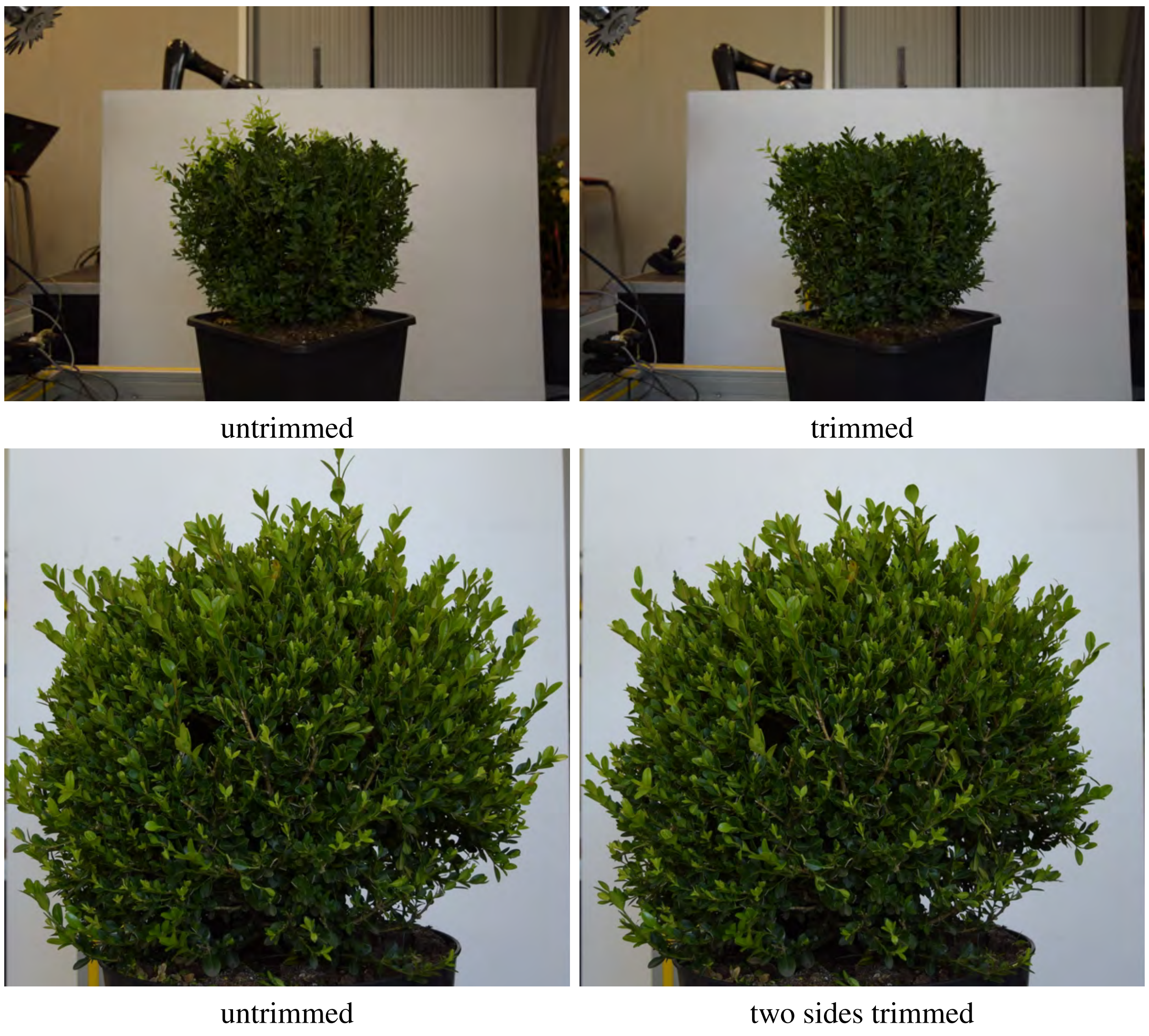


Figure 10: Top: Trimming evaluation of the cube bush with outgrowth. Bottom: Trimming evaluation of a spherical bush with significant outgrowth. After one trimming step, the bush was rotated by 90° and re-scanned. A new trimming trajectory was planned and executed, after which the bush was scanned once more. The top images show that outgrowth was removed on the left and right sides, but not all branches were caught by the trimmer.

- The vision module works as expected. The used hand-eye calibration method improved the overall accuracy and allowed performing multiple viewpoint shape fitting. However, it is desired to develop a more automated hand-eye calibration method. Shape fitting proved to work on a wide range of shapes.
- The path planning module provides feasible trimming paths. To improve trimming quality, additional constraints on the allowed and expected motions might be required. Also the pose of the robot with respect to the bush might have to be accounted for.
- The pipeline for automatic scanning, trimming, evaluation of trimming results and if necessary re-trimming works as designed. Trimming was only evaluated on spherical and cubical bushes. The trimming evaluation clearly showed a difference in 3D point distribution before and after trimming. Including automated correction of the trimming paths is future work.

3 Rose clipping evaluation

We have evaluated the static rose clipping scenario with the integrated system as it was presented at Demonstrator 2 event (Deliberable 7.6), ie. with 3D position based visual servoing to red markers placed on rose bush stems. In addition we evaluate the performance of individual vision components of the rose clipping pipeline (branch segmentation and bud detection) that will be integrated soon on the final platform. A demonstration video clip of rose bush clipping is available on the Trimbot2020 YoutTube channel [2].

3.1 Materials and Methods

We used an eye-in-hand configuration to make the experiments and evaluation. We mounted a stereo camera on top of the end-effector of the Kinova Jaco2 robotic arm, as described in deliverable 2.4. The robot has 6 degrees of freedom and it has a clipper mounted as end-effector. The setup and methods used are described below and generally follow D2.4, but some simplifications were done for the Demonstrator 2 because some of the vision modules could not be yet integrated and tested. While some of the components involved will be upgraded later, the evaluation methodology will remain the same.

3.1.1 Camera system for rose clipping

We used a stereo camera for the vision module. The camera has a resolution of 720×480 pixels and a baseline of 2.5 cm. The left camera provides colour images whereas the images captured by the right camera are in gray scale as seen in Figure 11.

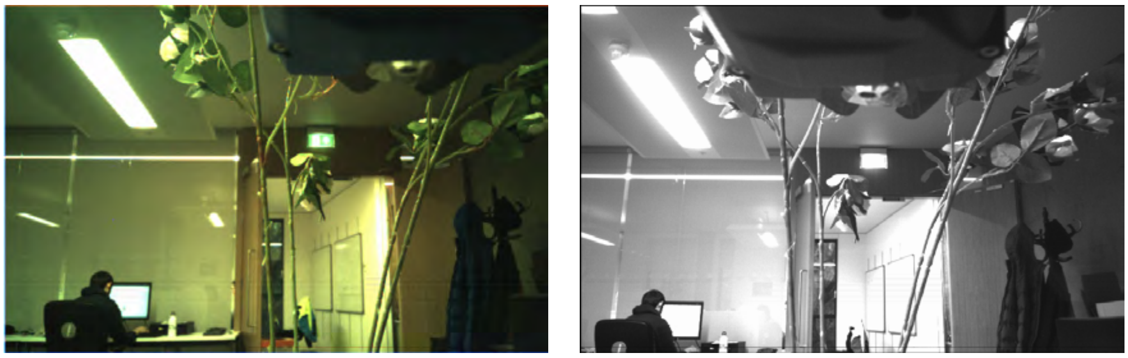


Figure 11: Images from the left and right camera respectively.

3.1.2 Vision module

To find the important parts of the bush and where to cut, we segment the bush from the rest of the environment and locate the bud-eyes.

For the first part, we use an encoder-decoder Convolutional Neural Network (CNN) with residual connections; a more detailed description of the architecture can be found in Table 1. The output of the network is a mask of the segmented bush.

Input image size:	256×256 px
Number of layers:	8
Filters per layer:	64
Kernel size:	3×3
Normalization type:	Standard
Equalization type:	HSV
Data augmentation:	10 %

Table 1: Branch segmentation CNN architecture.

For the second part, we find the bud-eyes in a rose branch by fine-tuning a pre-trained CNN detector YOLO [9]. To fine-tune the network, we replaced the last layer for a new output which consist of the coordinates of a bounding box and the probability of containing a bud-eye.

3.1.3 Visual servoing trajectory control

The visual servoing has the following steps. First, we perform a preliminary scanning of the rose bush to get the overall perspective of the bush we want to prune and to verify detections from multiple views. The scanning is done by moving the robot along the bush in a square trajectory to capture all the possible targets. While it scans, it collects the positions of the possible targets (XYZ points).

After the scan finishes, we check how close one point is to another. If the distance is less than a threshold, we merge the points by averaging their positions. We only keep the merged points

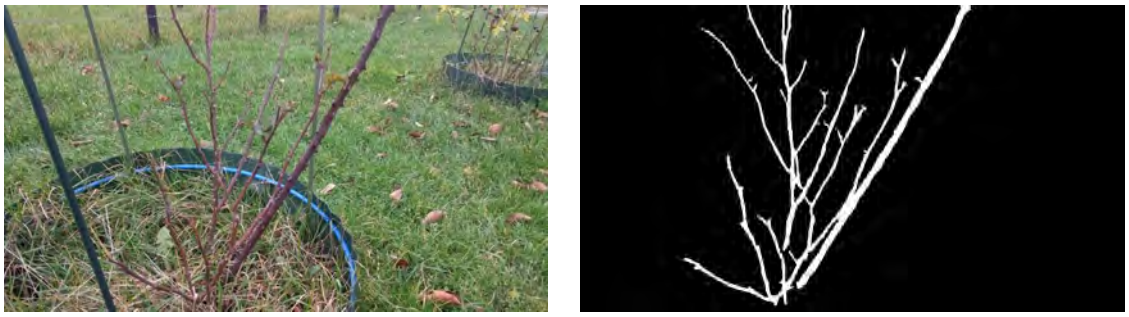


Figure 12: Example of the TB-roses dataset. On the left we have the colour image and on the right the ground truth mask.

with more observations than a given threshold. This means that if the vision module found a false positive target after processing an image in view t_i, it is less probable that it will find the same false positive target again in some other view t_{i+k}, leaving the false positive position (point) found in time t_i with only few observations.

Finally, we sort the stored positions of the targets according to the depth, i.e. the first target to be cut is the closest to the arm base.

We use position-based control to navigate towards the targets. While the end-effector is moving towards the target, we continuously run the visual module to update the position of the target. Once the clipper reaches it, we find the axis of the branch, rotate the clipper so it gets perpendicular to branch. Then, we go up 2 cm in direction of the branch, rotate 45° and send signal to clip the branch. We repeat the whole process until we have cut all the targets. Then we send the robot to a new section of the bush and repeat the scanning and navigation steps.

3.1.4 Dataset for evaluation

We used two datasets to evaluate the performance of the branch segmentation. One real called TB-roses [10] which consist of 100 images. A sample of the dataset can be seen in Figure 12.

We have also created a synthetic dataset which consist of 2880 images. These images corresponds to 80 different perspectives around 36 synthetic bushes in 4 different environments, as seen in Figure 13.

Finally, we created a dataset to train the bud detector by taking 134 images of different rose bushes in a natural environment. An example of the dataset can be seen in Figure 14.

3.2 Results

3.2.1 Branch segmentation

We trained and tested the network using the real and synthetic datasets mentioned in Section 3.1.4. For both, we used k-fold cross-validation for evaluation. The real dataset has 100 images

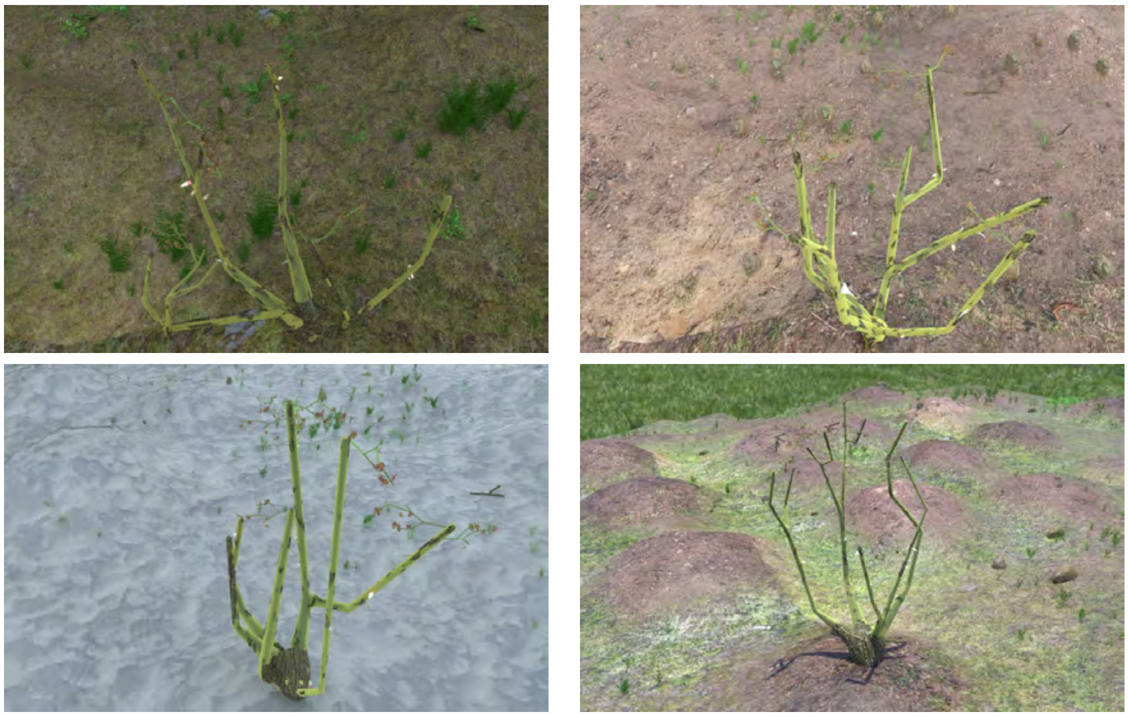


Figure 13: Example of the synthetic dataset. One bush per environment.

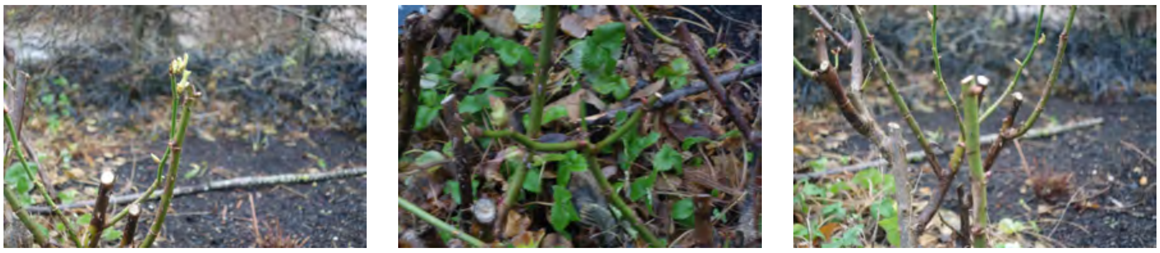


Figure 14: Example of the bud-eye dataset.

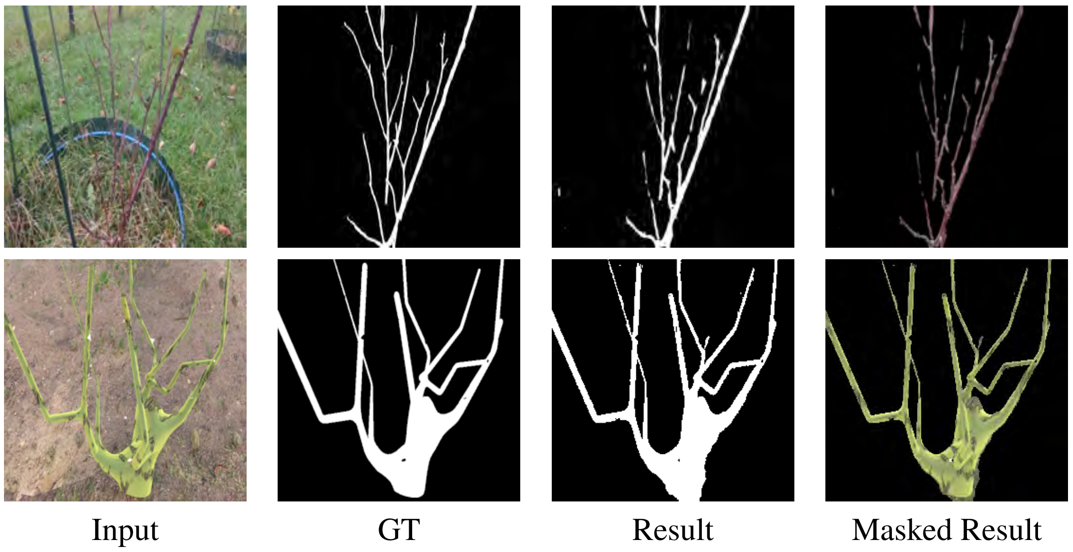


Figure 15: Segmentation of the branches for the real and synthetic dataset. The first row shows the results using the real dataset and the second row shows the result of the network trained with the synthetic dataset. From right to left, the first column shows the input image, the next column shows the ground truth. the third column shows the inference from the network, the final column shows the input image masked.)

and we used 3-fold cross validation. The data was split in 40%, 40%, and 20%. The synthetic dataset consist of 2880 images. Because we have 4 different environments in the dataset, we used 4-fold cross-validation to evaluate the network. This means we trained the network using 3 environments and tested on the one that is left. We used F1 score as a metric for the cross-validation. The results can be observed in Table 2 and the output of the network is shown in Figure 15.

Dataset	F_1
Real dataset	72.03±4.12
Synthetic dataset	88.70±5.98

Table 2: Result of the segmented branches for the real and synthetic dataset.

3.2.2 Bud detection

We trained the network using 107 images and tested on 27. The images were taken from different rose bushes, from different perspectives, light variation and in a natural environment. We use AP as metric and also report the precision, recall and F1 score. We report these values in Table 3 using an IoU threshold of 0.1 and 0.5. The output of the network is shown in Figure 16. Although the precision and recall seem low, we have to bear in mind that the dataset we used comes from an uncontrolled environment and the training set was small.

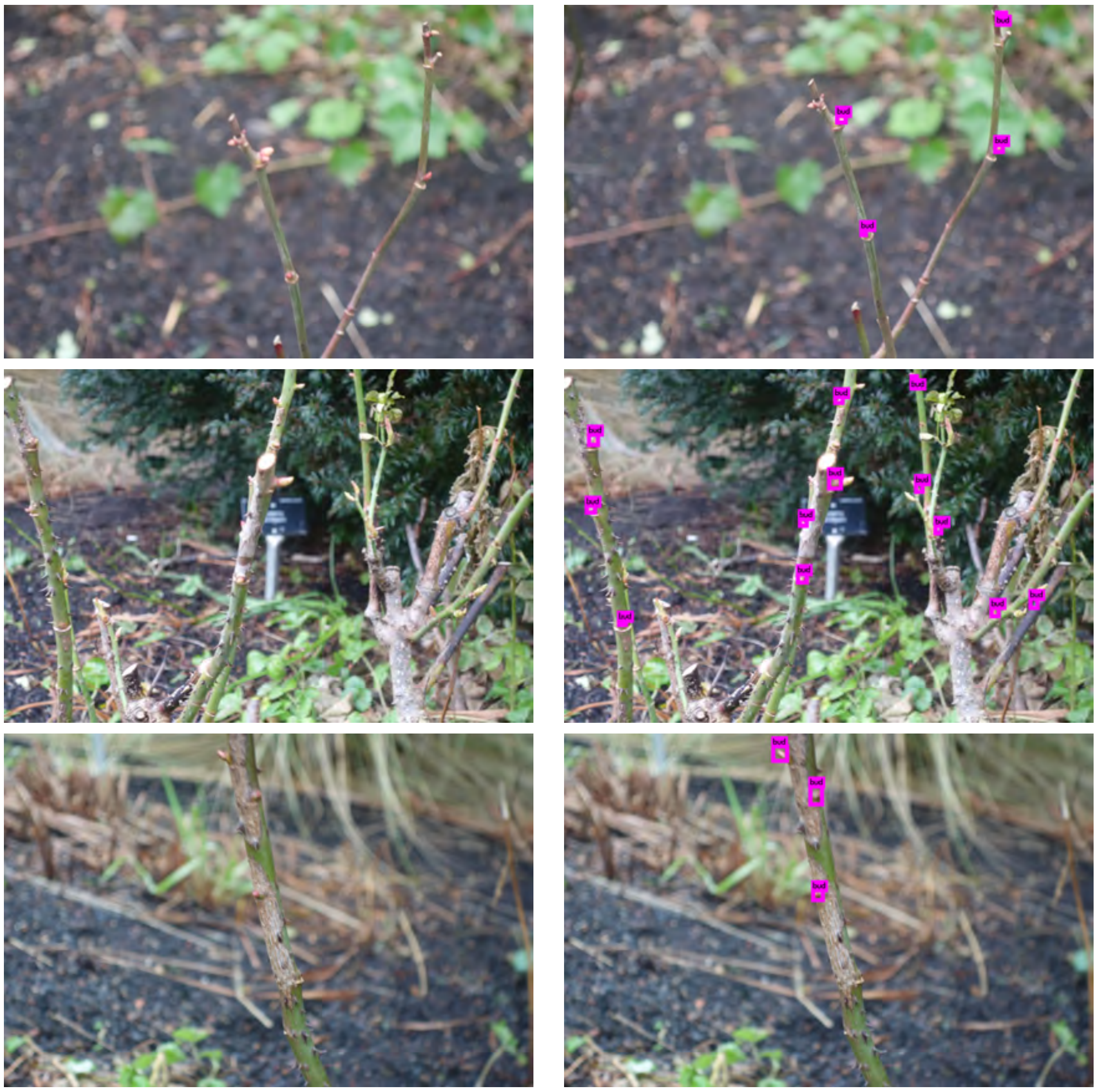


Figure 16: Inference of the bud detector. The left images are the inputs of the network and the right images are the outputs.

	AP	Precision	Recall	F1
IoU (0.1)	0.51	0.68	0.37	0.48
IoU (0.5)	0.17	0.34	0.19	0.24

Table 3: Result of bud detection with different IoU.

3.2.3 Clipping sites from scanning

We measure the accuracy of the scanning by comparing the total number of targets found after scanning a bush and the real number of targets. We also compare this value against the number of targets found by evaluating only one frame. The scanning captures colour images from which the targets are segmented and their center of gravity found. Each image is taken by moving the robotic arm in a square shape of $20cm$, as seen in Figure 17. Because we have a calibrated stereo camera, these points are transformed into a common coordinate frame which is the base of the robot. Finally these points are clustered by proximity of $0.5cm$ and then merged.

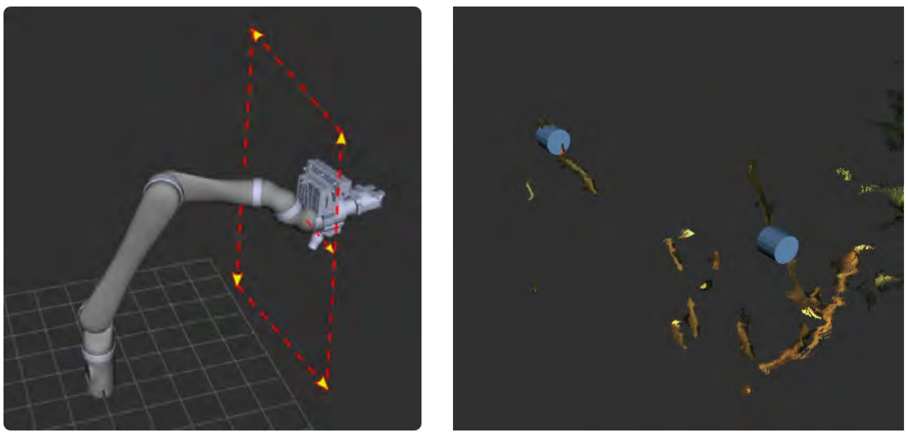


Figure 17: Left: Box shape scanning pattern. Right: Fused 3D result showing in blue the targets found after the scanning.

The evaluation was the following: We scanned a bush, each scan took 18 individual images where the targets were segmented and fused. We compared the total number of targets found after the fusion and by each of the 18 individual images. This process was repeated 35 times, with 8 different bushes from different perspectives. The experiment shows that if the targets are not fused, the accuracy is 61.31% and false positive rate is 15.23%. Whereas, if the scanned targets are fused, the accuracy of the method improves to 98.44% and the are no false positives.

3.2.4 Visual servoing and rotation maneuver

We evaluated the quality of the visual servoing system by placing red markers randomly across a rose bush and letting the robot navigate towards them. We repeated the experiment 35 times. We count as a successful navigation if the robot reaches the red marker and the stem gets into

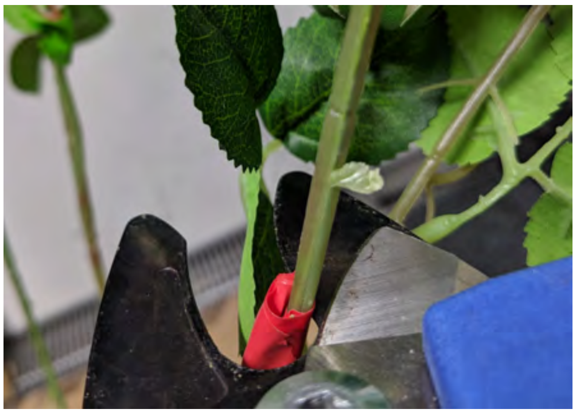


Figure 18: Result of the visual servoing where the end-effector reached the target with the tool in the centre of the target.

the clipper. The experiment shows that 96.82% of the time the clipper successfully reaches the target. We can see in Figure 18 the result of a successful servoing

Beginning and end of maneuver

We evaluate the subsequent rotation maneuver by measuring the angle of the robot's end-effector with respect to the branch prior to cutting. If the end-effector is perpendicular to the middle axis of the branch with tolerance of 10 degrees, we count it as a success maneuver, otherwise we report failure. Our method successfully performs the maneuver 92.86% of the times.

3.2.5 Stem clipping

We evaluate the clipping system by a series of clipper activation with stems of varying diameter inside. We report success if the stem is successfully cut in two parts, or failure due to mechanical, electrical or software error of the controller. The experiment consist of clipping 118 branches with different diameter and positions. The result of the experiment showed that the clipper cut all the 118 branches successfully. However, if a branch does not get completely inside the clipper, it only cuts the branch partially.

3.3 Discussion and Conclusions

In section 3.2.2 we can observe that the precision and recall of the bud detector is quite low. However, it is important to highlight that the data used for training and testing comes from an uncontrolled and complex environment, as seen in Figure 16. Another point to notice is that the bud detector network was fine-tuned with a small training dataset. More data will be collected and labeled in order to make a more reliable and general network for bud detection.

Currently, the rotation maneuver accuracy depends on the quality of the depth image which is noisy and not reliable for thin branches. This happens because the block matching algorithm used to compute the disparity images barely finds some correspondence between the two stereo images. To improve the rotation maneuver accuracy, we will rely on the colour segmentation instead, which obtains a good performance as seen in table 2.

4 Appendix

This deliverable has two appendices that are currently only available for members of the Trimbot2020 consortium:

4.1 Appendix 1

Unpublished manuscript: Coverage Trajectory Planning for a Bush Trimming Robot Arm [5]

4.2 Appendix 2

Unpublished manuscript: Automated Boxwood Topiary Trimming with a Robotic Arm and Integrated Stereo Vision [6]

References

[1] Dmitry Chetverikov, Dmitry Svirko, Dmitry Stepanov, and Pavel Krsek. The trimmed iterative closest point algorithm. In *Object recognition supported by user interaction for service robots*, volume 3, pages 545–548. IEEE, 2002.

[2] Trimbot2020 Consortium. Video of bush clipping demonstration (milestone 5): Rose bush clipping, 2019. URL: https://youtu.be/O2Tk09NYSXc.

[3] Trimbot2020 Consortium. Video of bush clipping demonstration (milestone 5): Topiary bush trimming, 2019. URL: https://youtu.be/W4UWhsn5X80.

[4] Eddy Ilg, Tonmoy Saikia, Margret Keuper, and Thomas Brox. Occlusions, motion and depth boundaries with a generic network for disparity, optical flow or scene flow estimation. In *European Conference on Computer Vision (ECCV)*, 2018. URL: http://lmb.informatik.uni-freiburg.de/Publications/2018/ISKB18.

[5] D. Kaljaca, B. Vroegindeweij, and E. van Henten. Coverage Trajectory Planning for a Bush Trimming Robot Arm. *Unpublished manuscript, submitted to Journal of Field Robotics*, 2018.

[6] Dejan Kaljaca, Nikolaus Mayer, Bastiaan Vroegindeweij, Angelo Mencarelli, Eldert van Henten, and Thomas Brox. Automated boxwood topiary trimming with a robotic arm and integrated stereo vision. *Unpublished manuscript, submitted to IROS 2019*, 2019.

[7] Jérôme Maye, Paul Furgale, and Roland Siegwart. Self-supervised calibration for robotic systems. In *2013 IEEE Intelligent Vehicles Symposium (IV)*, pages 473–480. IEEE, 2013.

[8] Nikolaus Mayer, Eddy Ilg, Philip Häusser, Philipp Fischer, Daniel Cremers, Alexey Dosovitskiy, and Thomas Brox. A large dataset to train convolutional networks for disparity, optical flow, and scene flow estimation. In *IEEE International Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016. URL: http://lmb.informatik.uni-freiburg.de/Publications/2016/MIFDB16.

[9] Joseph Redmon, Santosh Divvala, Ross Girshick, and Ali Farhadi. You only look once: Unified, real-time object detection. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 779–788, 2016.

[10] Nicola Strisciuglio, George Azzopardi, and Nicolai Petkov. Brain-inspired robust delineation operator. In *European Conference on Computer Vision*, pages 555–565. Springer, 2018.