TrimBot2020 Deliverable D3.3

Integrated 3D extraction package

Principal Author: ETH Zürich (ETHZ)
Contributors: Albert-Ludwigs-Universität Freiburg (ALUF), University of Edinburgh (UEDIN)
Dissemination: CO

Abstract: This report briefly describes the integrated software package combining the components from D3.2, providing the analysed scene data needed by the motion planning, navigation and hedge trimming control packages.

Deliverable due: Month 32

1 Overview

This report accompanies the software packages containing the components described in Deliverable D3.2, i.e., the SLAM and localization system and the 3D processing packages. The report is intended to provide an overview of the software, which is stored in the Gitlab repository of the TrimBot2020 project. For an algorithmical description of the individual components and their experimental evaluation, please see Deliverable D3.2.

The software described in this deliverable consists of two parts: The **SLAM** package is responsible to build a sparse 3D map of the garden and to localize the TrimBot in the garden using this map. The resulting robot poses, as well as the images captured by the cameras mounted on the TrimBot, form the input to the **3D Data Processing** package. This package computes depth maps from the input images and generates dense 3D models of the garden. In the following, we provide a brief overview over the two packages. More details can be found in the `README` files provided by the individual components. For a detailed description of the messages exchanged by the individual components, please see Deliverable D3.1.

2 SLAM

The `SLAM` package (see Sec. 2 in Deliverable D3.2) is available in the TrimBot2020 gitlab repository under `TrimBot2020/SLAM`[1]. The package consists of two parts: `gcslam` (Generalized Camera SLAM) is the actual implementation of the SLAM system, including the localization sub-system. `gcslam_ros` implements a ROS [1] node that provides the input to `gcslam` and provides the camera poses and 3D map estimated by `gcslam` to other processes such as the 3D Data Processing package.

`gcslam` is based on the COLMAP [2–4] Structure-from-Motion library. The package provides the options to either create a 3D map from scratch or to use an existing map for localization (by setting the `map_path` flag). In addition, it provides various flags that control the parameters of the SLAM and localization algorithms, including an option for high-quality (offline) reconstruction via the `--offline_reconstruction` parameter and an option for visualization (via the `--visualization` flag). Detailed instructions on how to compile, configure, and launch `gcslam` in the context of the TrimBot2020 project are provided in the `README` file of the software package.

The `gcslam_ros` package includes the `gcslam` package and acts as a wrapper. Besides providing automatic compilation of the `gcslam` system, the package integrates the `gcslam` pipeline into the ROS system used in the TrimBot2020 project. The package provides multiple launch files, including files for online operation (`run_live.launch`) and processing bag files (`run_bagfile.launch`). Instructions on installing the package and the package's parameters are provided in its `README` file.

For simplicity we collect all code in the `SLAM` repository, which contains both the ROS wrapper of `gcslam_ros` and the `gcslam` package. This is intended to be compiled as the `gcslam_ros` ROS node, but `gcslam` can still be compiled independently and without ROS by following the instructions in the respective subfolder's `README` file.

[1] `https://gitlab.inf.ed.ac.uk/TrimBot2020/SLAM`

3 3D Data Processing

The `3D Data Processing` package is available in the TrimBot2020 gitlab repository under `TrimBot2020/Proc3D`[2]. The `3D Data Processing` package provides different functionality used by the components from Deliverable D3.2:

- `SceneFlowALUF`[3] and `StereoALUF`[4] provide functionality for estimating scene flow and disparity maps from images. The packages include the "DispNet" neural network architecture [5] used to estimate disparity maps from stereo images (see Sec. 3.2 in Deliverable D3.2).
- `InstantFusion`[5] implements a fusion algorithm that merges disparity maps estimated by DispNet and the FPGA SGM stereo approach.
- `LocalFusion`[6] provides functionality to register dense 3D point clouds estimated by the sensing capabilities of the robot. This package is intended to provide both a set of robot poses and a dense 3D point cloud of the garden (see Sec. 4.2 in Deliverable D3.2).
- `GlobalFusion`[7] implements a volumetric fusion approach that takes multiple disparity maps (and their corresponding known poses) and integrates them into a volumetric garden representation. `GlobalFusion` also provides functionality for extracting a 3D mesh from the volumetric representation (see Sec. 4.3 in Deliverable D3.2).

The individual parts will be discussed in the following.

3.1 SceneFlowALUF and StereoALUF

A detailed description of the functionality and the software package was provided in Deliverable D5.2.

3.2 InstantFusion

The `InstantFusion` provides three ROS nodes that perform the fusion of FPGA stereo and DispNet disparity maps:

- `edin_sfgan_super` implements the SDF-MAN approach [6] that uses a deep neural network to fuse both types of disparity maps.
- `edin_disparity_view` produces colored disparity maps intended for visualization.
- `edin_instant_fusion_pipeline` provides a launch file for the complete fusion pipeline.

[2] `https://gitlab.inf.ed.ac.uk/TrimBot2020/Proc3D`
[3] `https://gitlab.inf.ed.ac.uk/TrimBot2020/Proc3D/tree/master/SceneFlowALUF`
[4] `https://gitlab.inf.ed.ac.uk/TrimBot2020/Proc3D/tree/master/StereoALUF`
[5] `https://gitlab.inf.ed.ac.uk/TrimBot2020/Proc3D/tree/master/InstantFusion`
[6] `https://gitlab.inf.ed.ac.uk/TrimBot2020/Proc3D/tree/master/LocalFusion`
[7] `https://gitlab.inf.ed.ac.uk/TrimBot2020/Proc3D/tree/master/GlobalFusion`

The package subscribes to the image stream of the multi-camera system mounted on the base of the robot and the disparity maps estimated by FPGA stereo and DispNet. It publishes the fused disparity maps (`local_fusion_disparity`) and their uncertainties (`local_fusion_disparity_uncertainty`). In addition, it publishes an image stream (`local_fusion_vis`) intended for visualization.

3.3 LocalFusion

The `LocalFusion` package implements the DUGMA algorithm [7] that is able to temporally fuse dense point clouds through rigid alignment. The package implements three ROS nodes:

- `edin_pcl_reg_dugma` implements the actual DUGMA approach.
- `edin_dispmap_to_pcl` provides functionality to convert disparity maps into the point clouds used by DUGMA.
- `edin_local_fusion_pipeline` provides a launch file for the complete pipeline.

The package subscribes to the images provided by the camera system as well as the point clouds provided by the `InstantFusion` package. It publishes the resulting aligned point clouds (`local_fusion_pcl2`) and the poses defining the alignment (`local_fusion_rel_pose_dugma`).

3.4 GlobalFusion

The `GlobalFusion` package implements a multi-scale fusion of disparity maps with known associated poses into a volumetric scene representation. The package subscribes to fused local point clouds and poses (`local_fusion_pcl2`, `local_fusion_rel_pose_dugma`) and global localization poses (`gcslam_pose`). The package publishes both the volumetric representation (`global_fusion_volume`) and a 3D mesh extracted from the volume (`global_fusion_surface`).

3.5 Additional Functionality

The `3D Data Processing` package also provides additional functionality used in Deliverable D3.5 but not in D3.2:

- `PoseFusion`[8] implements an Extended Kalman Filter (EKF) for fusing the robot pose estimates provided by `gcslam`, wheel odometry, and IMU integration.
- `aluf_deeptam`[9] implements the DeepTAM algorithm used for tracking the motion of the arm as part of Workpackage 4.

Details on both the Pose Fusion and DeepTAM are provided in Deliverable D3.5.

[8] `https://gitlab.inf.ed.ac.uk/TrimBot2020/Proc3D/tree/master/PoseFusion`
[9] `https://gitlab.inf.ed.ac.uk/TrimBot2020/Proc3D/tree/master/aluf_deeptam`

References

[1] Quigley, M., Conley, K., Gerkey, B.P., Faust, J., Foote, T., Leibs, J., Wheeler, R., Ng, A.Y.: ROS: an open-source Robot Operating System. In: ICRA Workshop on Open Source Software. (2009)

[2] Schönberger, J.L., Frahm, J.M.: Structure-From-Motion Revisited. In: CVPR. (2016)

[3] Schönberger, J.L., Zheng, E., Pollefeys, M., Frahm, J.M.: Pixelwise view selection for unstructured multi-view stereo. In: European Conference on Computer Vision (ECCV). (2016)

[4] Schönberger, J.L., Price, T., Sattler, T., Frahm, J.M., Pollefeys, M.: A vote-and-verify strategy for fast spatial verification in image retrieval. In: Asian Conference on Computer Vision (ACCV). (2016)

[5] Mayer, N., Ilg, E., Häusser, P., Fischer, P., Cremers, D., Dosovitskiy, A., Brox, T.: 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) arXiv:1512.02134.

[6] Pu, C., Song, R., Li, N., Fisher, R.B.: SDF-MAN: Semi-supervised depth fusion with multi-scale adversarial networks. arxiv **under review** (2018)

[7] Pu, C., Li, N., Tylecek, R., Fisher, R.B.: DUGMA: dynamic uncertainty-based gaussian mixture alignment. In: Proc. 3DV. Volume abs/1803.07426. (2018)

