3DSSC (3D Smart Sense and Control)

Deliverable D2.1

Software Architecture

3D Smart Sensing and Flexible Task Programming for On-Line Trajectory Adaptation in Fast Surface Treatment

Karel Vander Elst (Flexible Robotic Solutions, Belgium)
Keivan Zavari (Flexible Robotic Solutions, Belgium
Aertbeliën Erwin (KU Leuven University, Belgium)
De Schutter Joris (Flexible Robotic Solutions, Belgium)
Delforge Philippe (Flexible Robotic Solutions, Belgium)

1 Architecture

In the current software architecture shown in Figure 1, the communication with the KUKA LWR robot is done via the Fast Research Protocol (FRI, top left) where the joint positions are communicated. This however will change in the future versions where the system is tested with an industrial robot.

eTaSL (top right) is used to solve the optimization problem of calculating the optimal robot joint commands. In order to connect the laser measurements to points in the reference frame, the forward kinematics of the robot are used and the laser measurement delays are compensated.

The laser measurements are then used in the surface tracker sub-block where the surface is modelled. The surface parameters are then sent to the eTaSL block where the robot control command is calculated in the form of desired joint velocities.

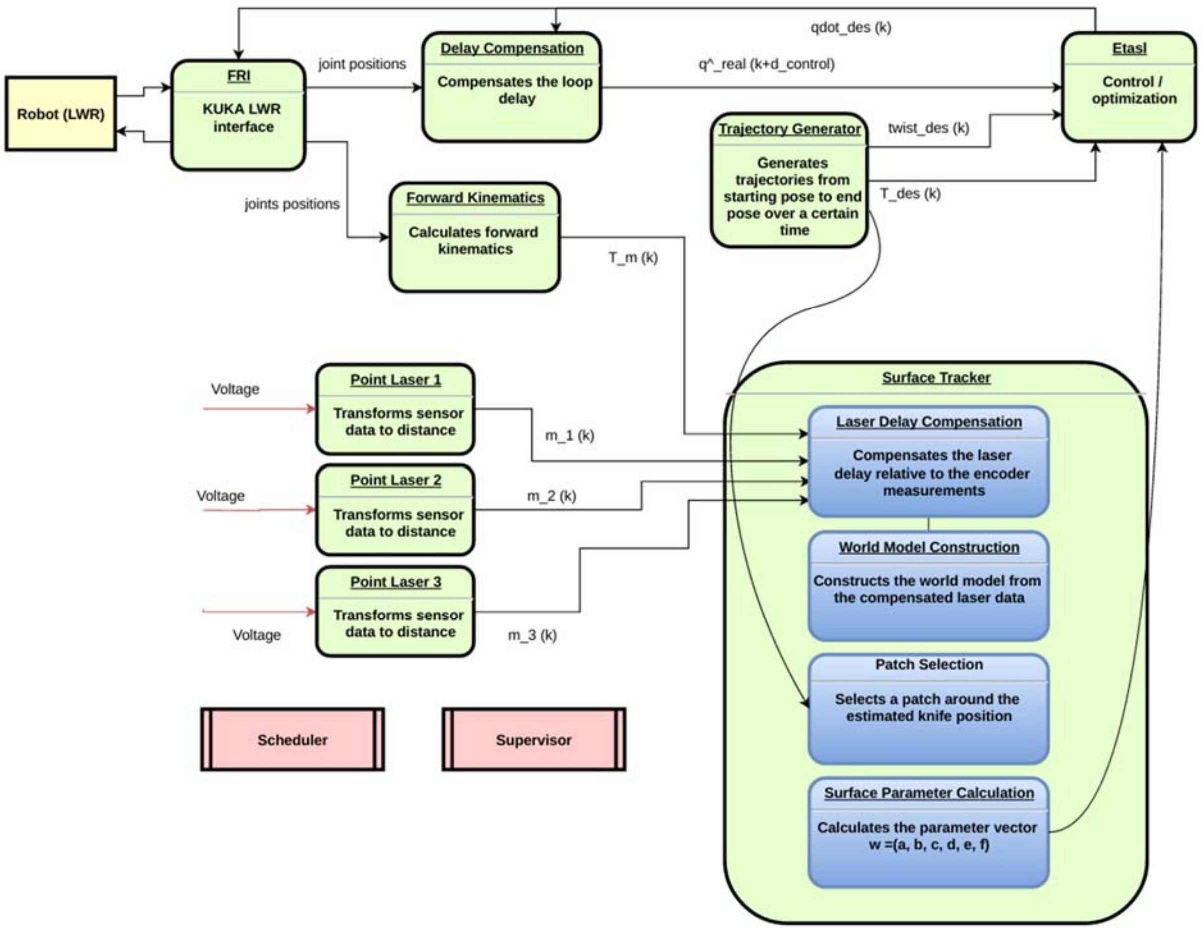

Figure 1: Software architecture of the surface modeling system

Delay Compensation

To achieve real-time accurate modelling of the surface of the workpiece, all incoming data need to be synchronized. The digitalization of the laser data creates a (hardware related) delay with respect to the robot encoder measurements. In other words: the laser data and robot position data received at a certain instance of time were not sent at the same time. To determine the location of the laser measurements with respect to a reference frame, this delay is compensated before the model construction is performed (see once again Figure 1).

Accurate modelling is however only the first half of the story. The delay between receiving the data and the robot acting on said data, limits the performance of the entire system. This is accounted for by calculating the surface parameters and control strategy for the expected position of the cutting tool, rather than its current position.

Trajectory Generation

The trajectory generation consists of two parts: the trajectory generator itself and the constraints defined in eTaSL. The trajectory generator creates a path between two points given a certain time frame. This path is sent to eTaSL, where the constraints are applied. As more and more measurements become available, these constraints gain more relative importance and the trajectory changes. For example, the robot can change the height of the tool to cut the measured local surface if the data is sufficiently reliable (otherwise the tool remains at a safe distance from the workpiece).

Surface Estimation

All laser data is, after delay compensation, added to the world model. In 'patch selection', depending on the position of the cutting tool, a subset of the data is considered for the local modelling of the surface, i.e. only the points on the surface close to the tool are relevant. These local data points are then approximated using a doubly curved quadratic function and the resulting six parameters are communicated to eTaSL.

Summary

The architecture has been implemented, is fully functional and has been tested in surface tracking experiments (i.e without actual cutting). These experiments were shown at the Hannover Messe (April 25-29 2016) on a KUKA LWR robot and ran well and robustly during the entire week. A demo video can be seen at https://goo.gl/tF7vI8.

References

[1] E. Aertbeliën, J. De Schutter, eTaSL/eTC: A constraint-based Task Specification Language and Robot Controller using Expression Graphs, proceedings of the 2014 IEEE/RSJ International Conference on Intelligent Robots and Systems, Chicago, 2014.