4SC020 - Embedded Motion Control

Initial Design Document

Rens Slenders - 1028611
Joeri Roelofs - 1029491
Matthijs van der Burgh -
Daniel Pijnenborg -
Sil Schouten - 0821521
Joep Linssen - 0815502

May 9, 2017

Introduction

The goal of this project is to autonomously solve a maze with a robot (PICO) as fast as possible. The robot has sensors, actuators and a computer on board to navigate through the maze. The computer runs on Linux 14.04 and the program language used for the software is C++. In this document the initial design is discussed, which consist of the requirements, functions, components, specifications and interfaces. This design can be used to make a structured software architecture.

Requirements

- Find exit.
- Find/open door.
- Avoid obstacles at all times.
- Complete the maze within 5 min.
- All tasks must be executed autonomously.
- System should work on any maze.
- Don't get trapped in a loop.

Functions

The functions which have to be implemented are subdivided into the different contexts. Motion functions describe how the software gets input from and supplies output to PICO. Skill functions describe how different skills are executed in the software. Lastly Task functions implement task scheduling.

The determined motion functions are:

- Basic actuation

Give PICO the ability to move around and rotate.

- Get information

Receive information from LRF and odometry sensors.

- Open door

Make a sound to open door.

The more advanced skill functions are:

- Mapping

Use information from "Get information" to determine and save a map of the environment. This creates the world model for PICO.

- Localisation

Determine where PICO is located in the map created in "Mapping". This will most likely be implemented in the same function.

- Object avoidance

Use current LRF data to make sure PICO does never hit a wall.

- Maze solving

Use the map in combination with current and previous locations within the map to determine a route out of the maze.

- Door and exit detection

Using the map to detect if possible doors have been found and if PICO has finished the maze.

The task control implements switching between the following tasks:

- Map making

Done throughout the challenge to improve current map

- Finding doors

Done in first stage of the challenge when no door has been found yet

- Finding the exit

Once a door has been found and opened the task is to go through the door and finish the challenge.

Components

To achieve the goal, it need to be split into different components. The four main components are world model, tasks, skills and motions. The world model creates a mapping of the maze and marks important features within it. When solving the maze, the robot logs the trajectory. The world model is not static and must be updated and combined with new data. The tasks of the robot are to open a door in the maze and to exit the maze. The task controller decides which task need to be preformed. To make a world model and to preform tasks, different skills are needed. SLAM (simultaneous localization and mapping) is used to construct and update the map of the maze. A maze algorithm is used to solve the maze as fast as possible. To navigate smoothly through the maze, driving skills and trajectory planning is needed. The robot must avoid objects at all times to prevent from being disqualified. To create motions, the robot has sensors to obtain data of its environment and actuators to move the robot in x, y and θ direction.

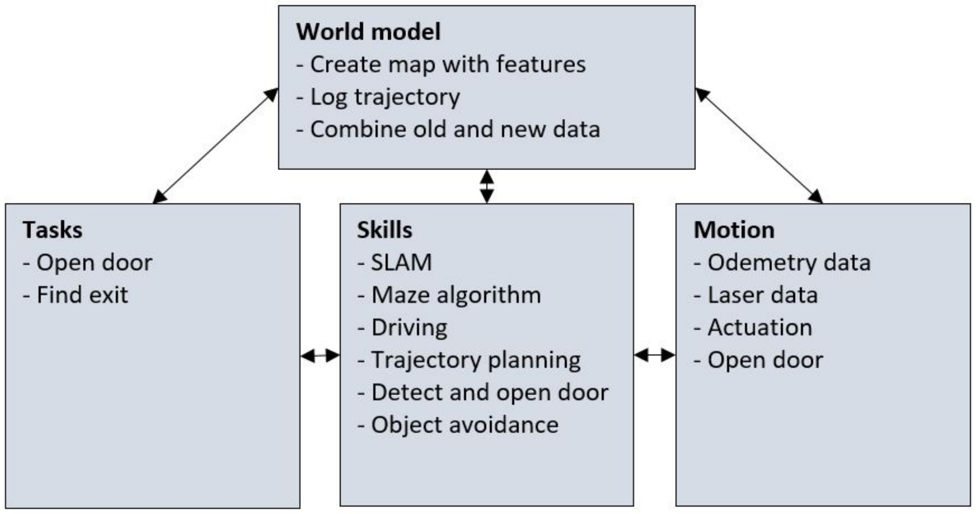

Figure 1: Interaction of all the components

Specifications

The following specification are quantified from the requirements

- The maximal velocity of PICO is limited to 0.5 m/s in any direction.
- The maximal rotational velocity of PICO is limited to 1.2 rad/s.
- PICO has to finish the corridor challenge within 5 min.
- PICO has to finish the maze challenge within 5 min.
- PICO cannot stand still for 10 sec or more.
- PICO should keep a minimum distance to the wall of > 20 cm.
- The mapping resolution should be ≈ 2.5 cm.

Interfaces

In order to test and debug the proposed algorithms in simulation, a visualization will be made. It is planned to contain the laser and odometry data, the features detected by the SLAM algorithm, the force velocity calculated by the Potential Field algorithm and the setpoints calculated by the maze solver algorithm. However, it will be made flexible, so that easily new visualizations can be added.