Department of Mechanical Engineering

Embedded Motion Control (4SC020)

Group 1

Toos van Gool	0885992
Paul Janssen	1273507
Jochem Manders	0858988
Max van Meer	0951669
Raoul Surie	0810262

Coordinator

Wouter Kuijpers Msc. w.j.p.kuijpers@tue.nl

Eindhoven, May 6, 2019

1 Introduction

The information structure proposed in this document is used to design the software of an autonomous robot, named PICO. PICO has to complete an escape room challenge and a hospital room challenge. To ensure good performance in these challenges, the requirements and specifications are defined initially. Afterwards, the hardware and software components are identified and the functions of the software components are defined. Finally, the interfaces between the components and functions are explained.

2 Requirements & Specifications

The requirements and specifications of PICO can be found in Table 1.

Table 1: Requirements & Specifications of PICO

Requirements	**Specifications**
PICO should execute all tasks autonomously.	Once PICO is started, no interaction is allowed.
PICO should perform all tasks without bumping into a wall.	The forward distance of PICO with respect to an object should always be at least 15 cm, sideways contact is not allowed.
Minimize oscilation of PICO to ensure correct sensor data.	PICO's acceleration profile should be smooth.
PICO cannot exceed speed limitations.	PICO's maximum translational velocity is 0.5 m/s. PICO's maximum rotational velocity is 1.2 rad/s.
PICO should be aware of its surroundings.	PICO should create and update a (local) map of its surroundings.
PICO should minimize its stationary time.	PICO should not stand still for longer than 30 seconds.
PICO should terminate when the objective is reached.	**ERC**: PICO should stop when its rear wheels have crossed the finish line.
PICO should fulfill its objective as fast as possible.	**ERC**: PICO should exit the room within 5 minutes.

3 Components

3.1 Hardware

The hardware of PICO consists of the following components

1. Sensors: Laser Range Finder (LRF), and wheel encoders (odometry)
2. Actuators: Holonomic base (omni-wheels)
3. Computer: Intel i7 running Ubuntu 16.04

3.2 Information

The information architecture of PICO consists of the following components

1. World model
2. Perceptor
3. Planner
4. Controller
5. Monitor

The world model contains the state of all activities on which the other components base their actions. The perceptor functions as a data processor that creates a perception of the world around PICO by interpreting the sensor data. The planner contains a state machine in which the strategy of a process is implemented and plans actions based on this state machine. The controller ensures that PICO executes tasks in a correct manner and the monitor ensures that problematic situations, such as encountering static or dynamic obstacles, are resolved. The manner in which these components communicate among each other is described in Section 5.

4 Functions

PICO will operate using a number of functions listed in Table 2. The functions are divided in three groups belonging to three software components.

5 Interfaces

The interfaces between the different components is illustrated in Figure 2. All components communicate with the world model to ensure that they operate using the same perception of surroundings and from the same tasks. This is realized by allowing the components to perform certain tasks based on the state of the world model. For example, if the planner notices that the exit is found in the escape room challenge, the state is changed to *ExitRoom*. The change of this state causes the controller, monitor and perceptor to only use the functions relevant to leaving the room. In this case, the functions of the controller could be *MoveToExit*, *MoveToTrajectory* and *AvoidStationaryObstacle*.

The world model thus contains the state on which the components base their events, current location with respect to a certain reference (e.g. the wall besides him or the position within a room), landmarks currently visible and past landmarks to determine the trajectory, a desired and actual trajectory and the incoming sensor data.

The planner contains both strategies for the challenges. The strategy to be used in the escape room challenge is illustrated in Figure 1. PICO starts in the *Orient* state to immediately find the exit if the exit is in sight, and otherwise a wall to follow. If a wall is found, the monitor indicates this by updating the world model. The planner notices this event and sets the state of the world model to *FollowWall*. This process continues until the state ObjectiveComplete is reached.

Table 2: List of functions used by PICO

Function	Description
Controller	
OrientInRoom	Determines whether towards move to an exit or a wall
DriveToWall	Drives PICO towards a predefined wall
DriveAlongWall	Drives PICO within a certain distance along a wall
MoveToExit	Drives to a point allocated in front of an exit
MoveThroughCorridor	Manoeuvers PICO through the middle of a corridor
AvoidStationaryObstacle	Adjusts trajectory to avoid stationary obstacle
AvoidDynamicObstacle	Adjusts trajectory to avoid dynamic obstacle
TerminateActivity	Shuts down PICO when objective is reached
MoveToTrajectory	Determines a feedforward trajectory to end up at desired trajectory
Monitor	
FindProximityToWall	Updates world model state to avoid objects
FindConcaveCorner	Adds new concave corner to the world model
FindConvexCorner	Adds new convex corner to the world model
TrackTrajectory	Updates world model that actual trajectory deviates too much from desired trajectory
FindStationaryObstacle	Updates world model with new stationary obstacle
FindMovingObstacle	Updates world model with new dynamic obstacle
FindExit	Updates world model state when two concave corners are found in close proximity
FindCorridor	Updates world model state when an exit with extending walls is found in close proximity
FindCabinet	Updates world model if cabinet is found
LostExit	Updates world model state if an exit is lost when moving towards one
LostWall	Updates world model state if a wall is lost when tracking one
Perceptor	
InitializeSensors	Ensures proper working conditions to operate
DeleteUnusedData	Deletes data which has become either irrelevant or unused
LocateWalls	

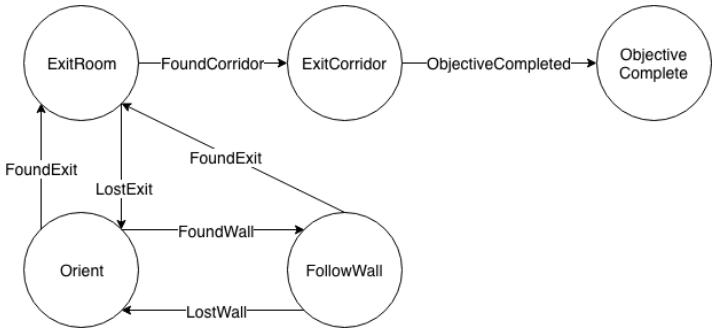

Figure 1: The proposed strategy of the escape room challenge

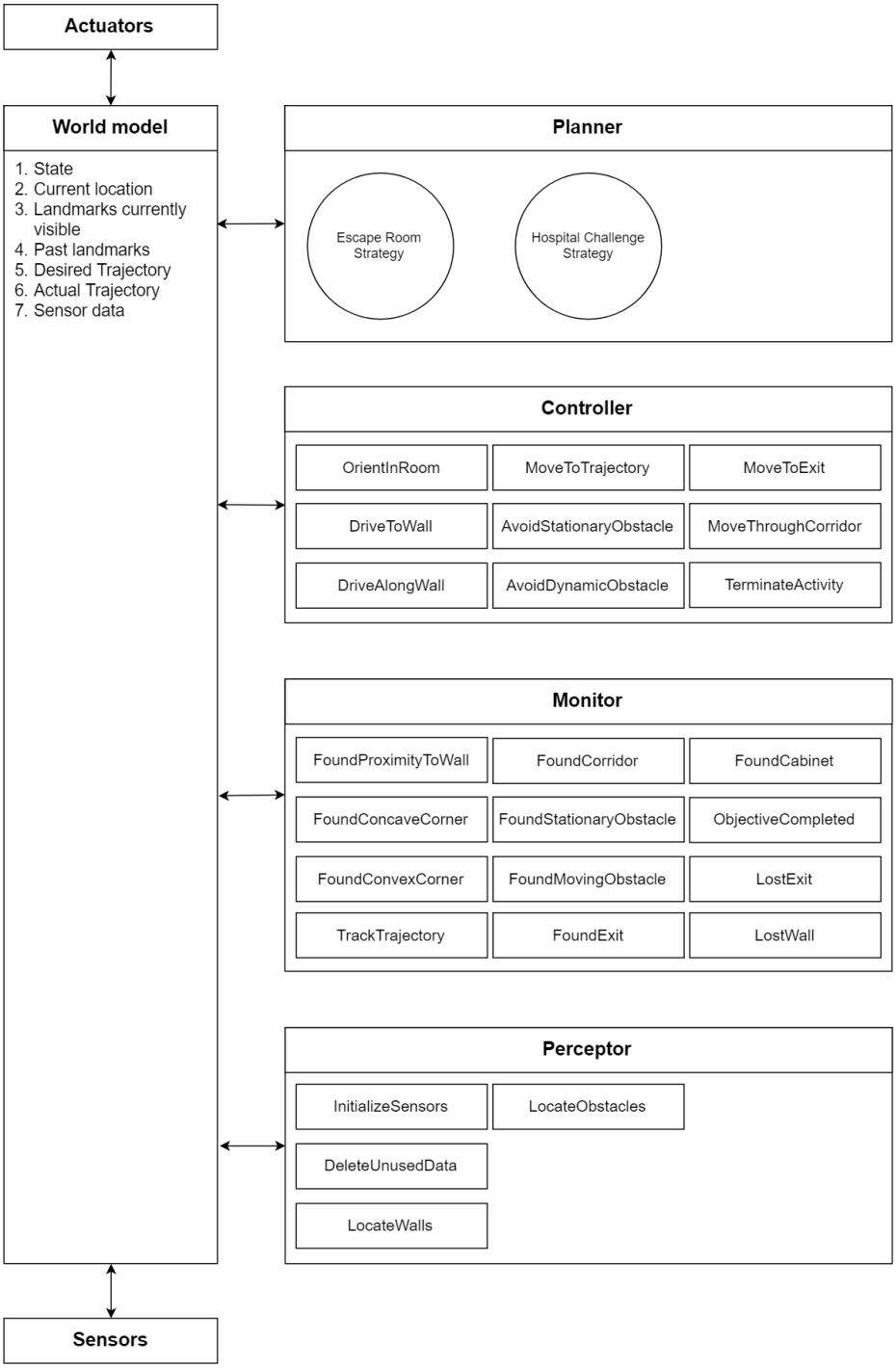

Figure 2: The proposed interface of information architecture