Introduction

motivation

Industry 4.0 strives to create so called "smart factories". In these the whole production starting from a request up to the finished product is one integrated process.

Warehousing is one part in the chain that brings a product to the consumer.

Automation saves time for people to do interesting stuff like automation

problem scenario

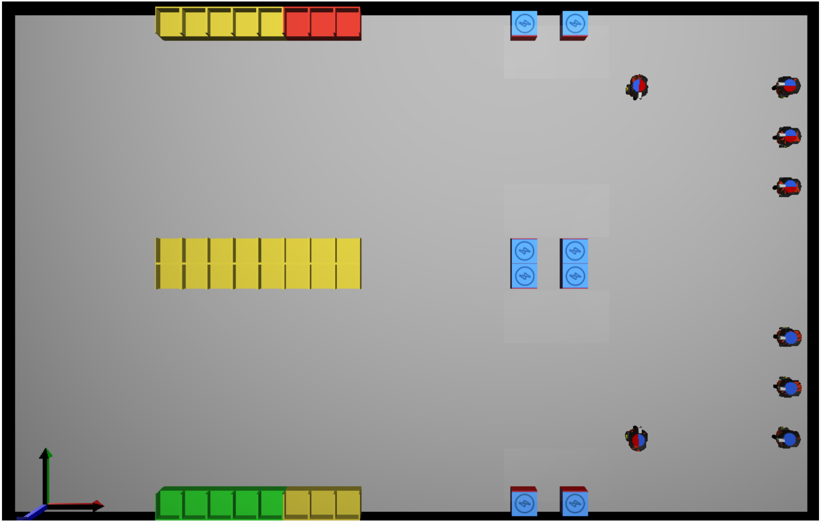
The typical scenario includes

A warehouse that has some kind of input and output, storage capacities and a map of all static obstacles.

Autonomous vehicles that can interact with the packages.

A dynamic environment that the vehicles have to perceive and evaluate to avoid collisions.

example of a medium sized map



problem definition

We defined several subproblems:

Task Planning

splits the work flow into tasks, manages them and assigns them to agents.

Path and Trajectory Planning

finds optimized paths between two points on a map and deals with dynamic obstacles.

Motion Planning

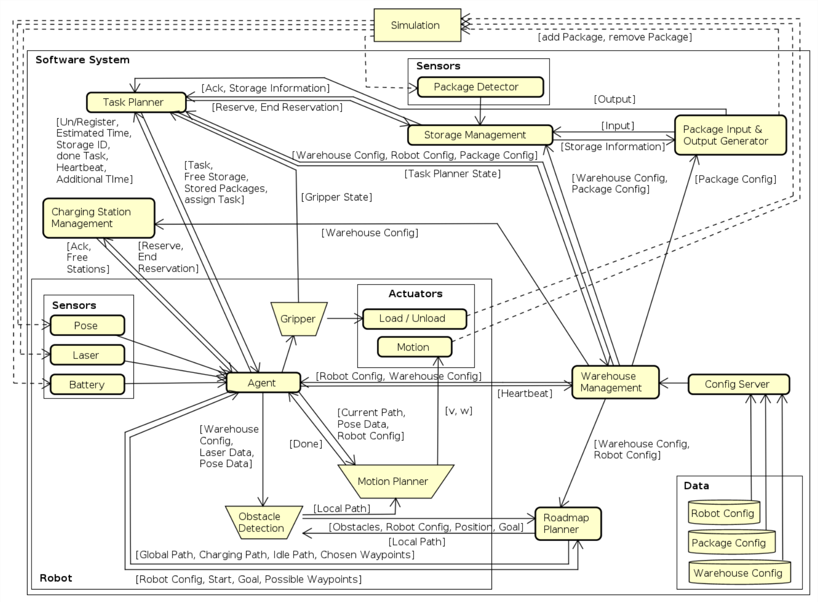
implements a model, which lets the robot drive.

Infrastructure

includes a Warehouse Management, an interface to the system and package request simulation

Methods

system architecture



task planner

Two stage architecture with a request layer and a task layer

Requests come from the outside and can either be of the types input or output

Tasks are multi threaded and one task tries to fulfill one request.

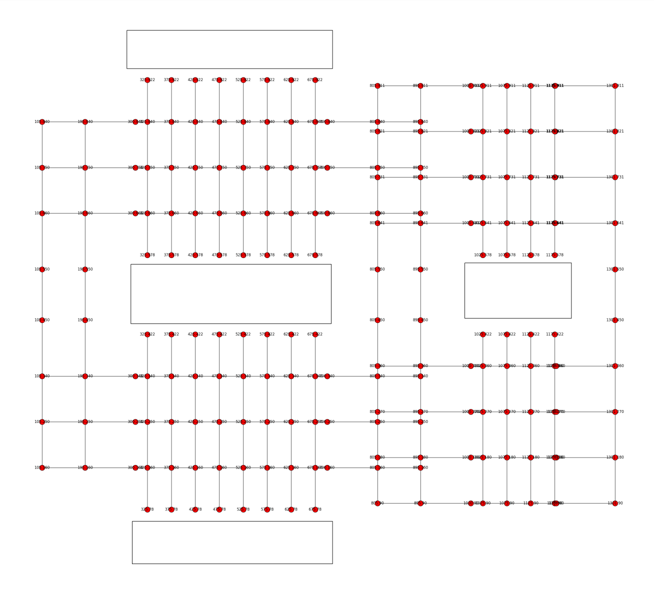
The planner cycles through all request and tries to meet them with free resources. Idle robots are queried for their nearest free tray and report back an ETA. The fastest robot gets the task.

roadmap planner

One roadmap per robot type is generated at the initiation of a simulation. The main steps are:

1. A euclidean distance transform of the map is calculated.
2. corridors are defined and filled with streets.
3. storage trays are connected to streets.
4. intersections are defined and streets are extended to connect.

roadmap planner example graph



local planner

The local planner uses local instances of the roadmaps and the sensory information of the agent.

1. The agent detects dynamic obstacles by comparing sensory data with the map of static obstacles
2. the local planner builds a 'forbidden zone' around each obstacle point.
3. edges in the roamap graph, that fall into the zone are deleted.

agent

The agent node combines several parts of our system:

Actuators: wheels and a gripper

Sensors: a laser array, a battery and a pose sensor

Obstacle detection by comparing sensor data with a map of static obstacles

Motion planning that lets the robot drive along a sequence of points

The rest of the nodes are auxiliary nodes that enable us to run a simulation:

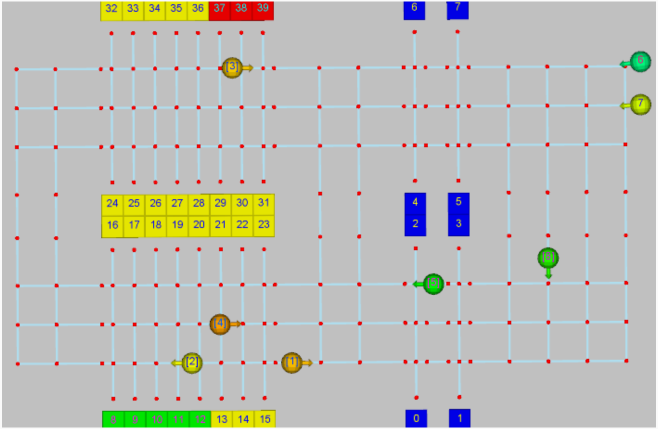
Morse is used to generate a simulation based on config files

Warehouse management initiates all other nodes

Storage management provides a central information pool about the state of our system as well as a visualization of the roadmap and the battery level.

Packages are handled by a generator that simulates requests for the task planner and moves packages from and to trays

visualization

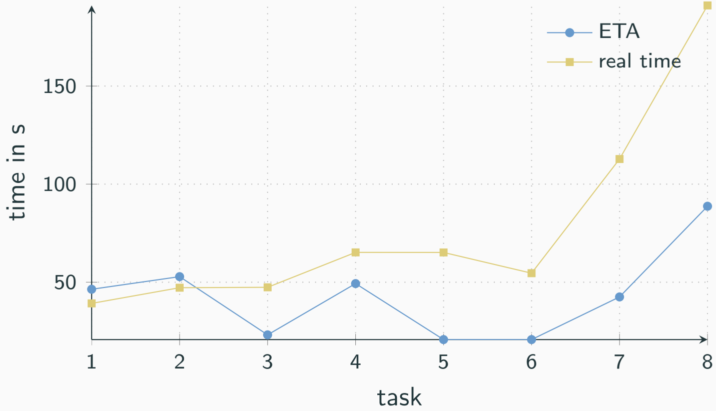


Results

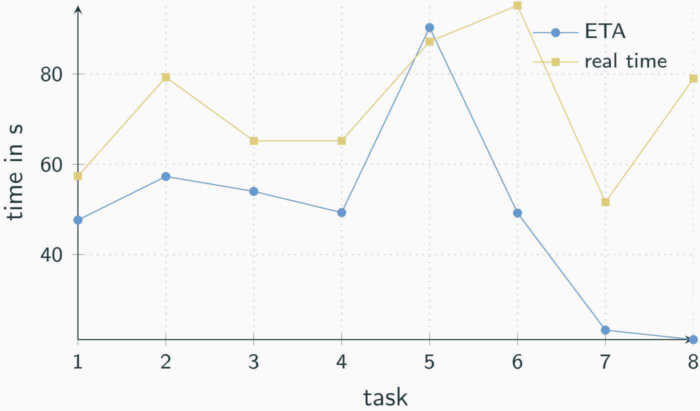
Table 1: Our Maps in detail

Map	Shape	# Storage	# Input	# Output	# Charging
1	8*8	16	1	1	0
2	14*10	32	3	3	8
3	20*20	100	5	5	20

results: eta vs real time on map 2 with 8 robots



results: eta vs real time on map 2 with 6 robots



discussion

Our solution is still a little error prone, and we need to collect more data to be able to analyze our solution regarding

Performance: We want to measure the optimal ratio between number of robots and request frequency.

Scalability: There is an optimal number of robots compared to map size. We want to find out, if this relation changes with the map size.

end

THANK YOU. QUESTIONS?