Robotic Drug Delivery System for Hospitals

Section 1. Introduction

Our project regards a robotic drug delivery system for hospitals.
We imagine a scenario in which a robot equipped with a trolley of unlimited capacity simulated by software, schedules and performs tasks of delivery of medicines to different patients according to their health needs. Our main focus is a time-saving planning of drugs and therapeutics delivery within a hospital department.

The delivery of drugs, in this context, is very important because of great help and support for health professionals who, by delegating this task to robots, will be able to focus at best on other types of problems.

Given the current emergencies, we wondered if some hospital in Italy already adopted a service of distribution of drugs via robots. We have learned that in the Morgagni Pierantoni Hospital in Forlì [1] [2] an automated method, employing an autonomous guided robots, was already in use. The system allows to move trolleys from one hospital department to another. This results in an improved organization of energy and time among the hospital staff as well as to lower the risk of error thanks to the greater standardization operated on the processes.

For the realization and simulation of our project we have been inspired by what is currently being done at the Morgagni Pierantoni Hospital in Forlì. Obviously we had to make some simplifications resulting from the use of Webots [3], a free and open source 3D robot simulator used in various fields, and the Platinum planning framework [4].

Despite some limitations, Webots gives the ability to develop, test and validate artificial intelligence algorithms in an easy way and allows to create or possibly modify existing robots.

The robot we have chosen is the Adept Pioneer 3-DX [5].

The Pioneer 3-DX is a small lightweight two-wheel two-motor differential drive robot ideal for indoor laboratory or classroom use. The robot comes complete with front SONAR, one battery, wheel encoders, and other functionalities. Pioneer research robots are the world’s most popular intelligent mobile robots for education and research. Their versatility, reliability and durability

have made them the preferred platform for advanced intelligent robotics. Pioneer robots are pre-assembled, customizable, upgradeable, and rugged enough to last through years of laboratory and classroom use.

The project is available on GitHub [6] under the GPL-3.0 license.

Section 2. Problem statement and requirements

The problem is as follows: for each patient a list of prescribed drugs is given. The patient should comply with time and the room in which the patient is hospitalized has to be taken into account. Therefore a scheduling that allows the delivery of drugs by the robot at the scheduled times, trying to optimize the routes and timing needs to be developed.

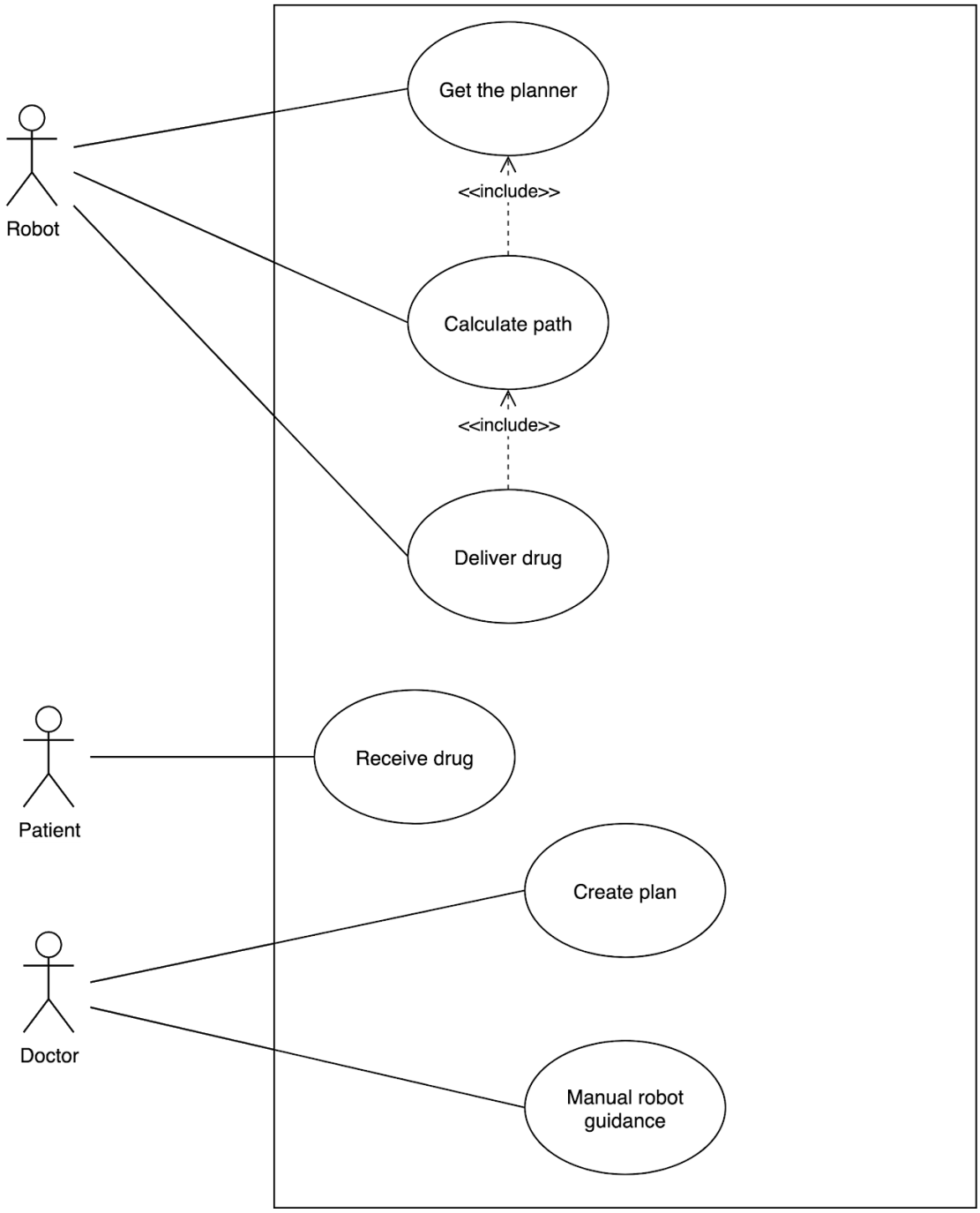
Since the entire project was developed within the Robot 3D Webots simulator, we had to make some simplifications based on the limitations of Webots.
In particular, after a careful and critical analysis of the simulator, we noticed some of its limitations, for example:

- it is not possible to fully simulate a human being, as this is only a simulation of a more generic Robot class
- integration with machine learning, at a good level of detail, is difficult and complex
- the creation of a sufficiently realistic scenario, requires different skills that are not part of the domain of competence of our problem

Therefore our problem has been simplified assuming that:

- the cart has unlimited capacity
- there is no interaction between robot and patient, but only between robot and hospital staff (planning and manual guidance)
- the medicines that every patient must take are already present in the cart

Below is the use case diagram of the problem:



The use case diagram represented highlights all the functions of the robot in relation to the doctor and the patient. In particular, the robot will consult the planner (PLATINUm) who will return the tasks, with their schedules, associated with the treatment plan provided by the doctor previously; the robot will calculate the optimal path to perform the tasks within the set time; the robot will deliver the drugs when it finds itself at the corresponding patient.
In addition, the doctor, at any time, can take full control of the robot through the "drive manually" mode.

The minimum requirements that the robot should have are:

1. distance sensors to avoid any obstacles present on his path
2. two-wheel and two-motor differential drive
3. a good speed as it must respect the time constraints for deliveries, moving in large and spacious environments (such as, for example, the long hospital corridors)

The ideal requirements, in addition to the previous ones, also include the size of the robot: having to carry a rather large "cart" containing several drugs, the robot should be sufficiently large and robust.

Section 3. Related Work

As reported in section 1, Morgagni Pierantoni Hospital in Forlì uses robots that allow for an automated movement of trolleys from one hospital department to another. In this case, they use eight autonomous robots which are entrusted with the handling of 4 different types of materials transported inside trolleys: waste, flat linen, medicines and meals. Each robot collects and delivers the trolleys according to a schedule received and based on a defined distribution platform. The routes are dedicated but not exclusive, the robots in fact correct their own or stop in front of obstacles that arise along their own race.
This is made possible by the complete mapping of the structure in the memory of each device, the position control calibrated by the Wi-Fi system and the laser technology that keeps the robots away from the walls.

Based on how this system was implemented in a real hospital, we also used the idea of mapping the structure of the entire hospital department and storing this information within our robot.
As for the position control, we preferred to use a GPS in combination with a compass, compared to Wi-Fi, as it seemed less expensive in economic terms and less constraining from a structural point of view. In fact, in this way our robot could be used in any hospital department even not equipped with Wifi system.

Subsequently, as done by the hospital in Forlì, we also created exclusive paths for the robot in order to avoid as much as possible to be an obstacle for hospital staff and patients. Finally, for any eventuality, a manual guidance system has been implemented that can be activated at any time by the competent staff.

This problem is part of a broader category of planning problems with time constraints. To solve this, we decided to use the Platinum framework that allows us to create planning models [9] [10] [11] [12].

For the creation of the model we have, initially, taken inspiration from some examples present within the Github repository of the project [4], until arriving at the final solution through subsequent refinements.

Section 4. Approach

In view of the minimum requirements and the different robot models offered by Webots, we opted for the Pioneer 3-DX. This model of robot has, by default, 16 sensors of distance to which we have added, inside, a gps and a compass, useful to calculate the paths based on the position of the robot within the simulated environment.

Compared to the ideal characteristics, our system presents limitation: for example it is not able to move a "cart". Considering that we are in a simulation in which we assume the presence of a "carriage" of unlimited capacity, this robot seems to be the ideal solution, thanks to its high speed.
In any case, the approach used could be generalized for any other robot that has at least the required components in the minimum requirements listed in section 2.

Algorithms

The first problem we had to deal with was moving the robot, so that it could move autonomously from one point to another**.**

Autonomous movement between two points

Initially we started from the approach used for autonomous driving robots used in the hospital of Forlì: the robots follow straight lines drawn on the floor that specify the path to follow.

Not having available lines on the texture that represents the floor of the hospital and not being able to add them manually (intuitively), we exploited the information obtained through the use of the GPS and compass that we installed on our robot, as not present by default.

The GPS allows us, at any moment, to obtain the coordinates of the robot in the hospital, while the compass has been set in such a way as to obtain the angle between the "face" of the robot and the direction vector pointing towards the north. Therefore, at every moment the robot, knowing the point of arrival, is able to calculate the direction it must take to reach the destination and above all change the speed of the two wheels in order to be able to align with it.

Once we had good results with regard to moving the robot within the map by straight lines, we used an artificial intelligence algorithm to calculate the optimal path between any two nodes in the graph.

A*

In particular, the algorithm used is A*, a graph search algorithm that identifies an optimal path from a given initial node to a given goal node.
We started from a first level of abstraction which is that of moving the robot along a single segment. Then we made sure that the robot could move between the different rooms following the different segments. Finally, the final result was to allow the robot to move between the various patients following a pre-established plan.

To be able to abstract all these steps it was necessary to map the entire hospital department, creating a graph whose nodes are represented by the extremes of each segment and the arc is the segment itself. Each arc is considered bidirectional.

Therefore, to move between hospital rooms optimally, it is sufficient to apply algorithm A* from the position in which the robot is located to the position in which it wants to arrive simply using the names of the rooms and the patients.

To avoid any type of inconvenience that could occur in a real situation within a hospital, we decided to implement a manual guidance mechanism that can be activated by hospital staff at any time.

Drive manually

To activate this mode, just press the key ‘E’ in any moment of the simulation.
This will allow the user to drive the Robot simply by using the keyboard.
There is also a predetermined button ('P') that allows to know, in real time, the coordinates of the robot inside the hospital.

To do this, it was enough to initialize the keyboard as if it were an add-on to the robot so that it could be used from the outside at any time. The various controls will set the speed of the wheels so that you can rotate the robot in the desired direction.

At this point we have a system that allows us to move along arbitrary and optimal paths (between rooms and patients) at any time of the simulation. Therefore, the problem that is not to be solved is that of planning with time constraints.

In fact, the idea is to reach patients at the times set by the therapeutic plan created by the doctor.
In order to do this, we used the timeline-based Platinum scheduling and scheduling framework. This was very useful to us regarding the automatic generation of plans for robots, with a certain degree of autonomy.

PLATINUm

Through Platinum we can define two types of files:

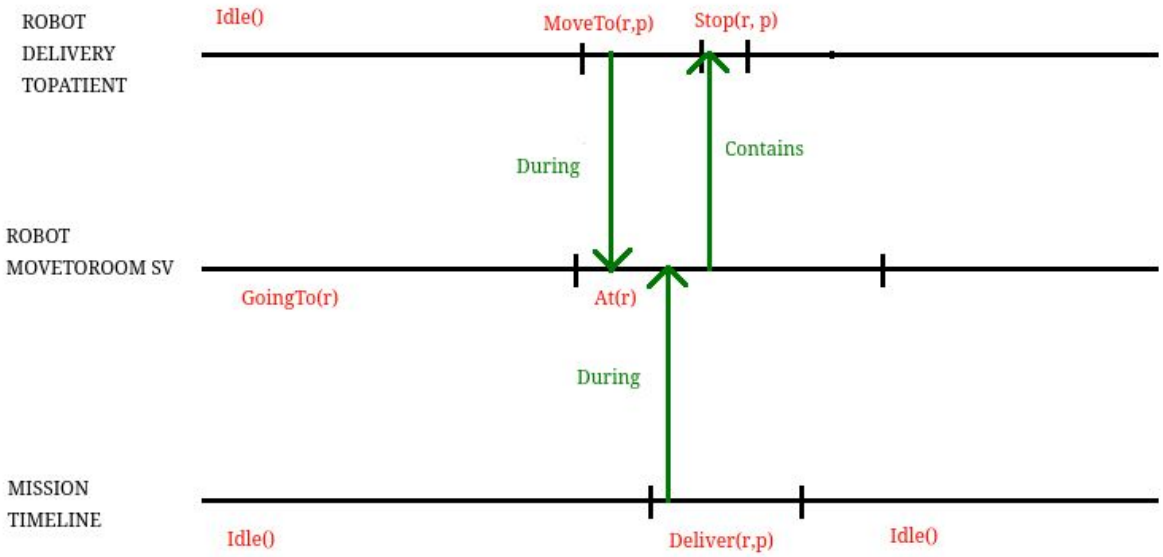
- **.ddl** which represents the domain of our problem
- **.pdl** representing instances of our domain

In the present case, the .ddl [8] contains information on status variables, the total duration of the timeline, and time and sync constraints.
The time window used is 2h, where each unit corresponds to 15 seconds. This allows us to obtain a certain elasticity for the generation of possibly more therapeutic plans to be distributed within the same day.
For our problem we used 3 status variables:

- RobotMoveToRoomSV, which can take on values:
    - GoingTo(room), to model the displacement between one room and another. We have modeled a duration between 5 and 30 units, i.e. between 01:15 and 07:30 min
    - At(room), to identify the presence of the robot in a certain room
- RobotDeliveryToPatient, which can take on values:
    - Idle(), when it doesn't have to do anything
    - MoveTo(room, patient), to model the movement between patients within the same room. Its duration is between 2 and 3 units, i.e. between 0:30 and 0:45 min
    - Stop(room, patient), to model the delivery of the drug to the patient. Its duration is between 1 and 3 units, ie between 0:15 and 0:45 min
- MissionTimeline, which can take on values:
    - Idle(), when it doesn't have to do anything
    - Deliver(room, patient), during the delivery of the drug to the patient

In addition, the MissionTimeline status variable allows the modeling of the goals specified in the .pdl.

The following is the diagram that defines the synchronization constraints used between the status variables:



It is possible to note that:

- RobotDeliveryToPatient **During** constraint for the value Moveto(room, patient) constrains the movement of the robot between patients of the same room only when the state variable RobotMoveToRoomSV assumes the value At(room) with unified room values
- RobotMoveToRoomSV's **Contains** constraint for the At(room) value constrains the RobotDeliveryToPatient status variable to take the Stop(room, patient) value if and only if the robot is present within the room room (with unified room values)
- MissionTimeline's **During** constraint for the value Deliver(room, patient) binds the delivery of the drug only when the robot is in the room

Within the file .pdl [8] are indicated some instances of the problem, which are divided into fact and goal. The fact is the initial condition of the problem (for example where the robot is located at the beginning of the simulation), while the goals represent the goals to achieve for which the planner will try to find the optimal solution. In our case they represent the delivery of medicines to patients on the basis of the specified times.

Once we got the file representing the calculated schedule from Platinum, we found ourselves faced with the problem of having to "translate" it into commands executable by our robot.

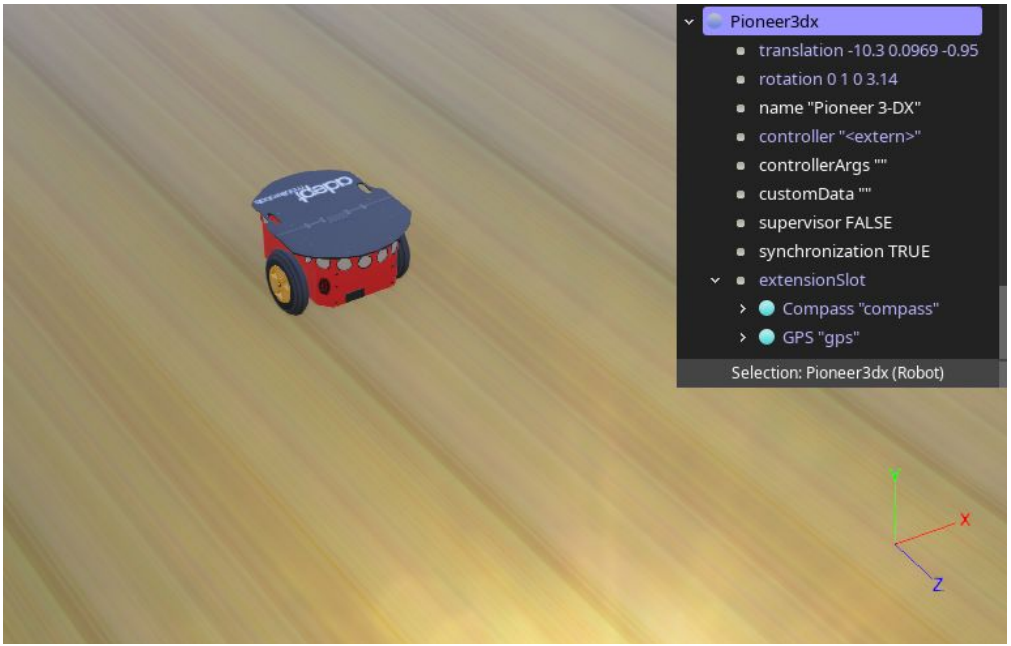
Plan integration with Pioneer 3-DX

To solve the problem, we have developed a parser that allows us to obtain the necessary information that allow the effective application of the plan by the robot.

To synchronize this information with the simulation environment we have introduced a global clock that allows us to coordinate the movements of the robot based on the times indicated in the schedule.

Below is the graph that was created and used to map the hospital:

The two yellow dots along the top right segment symbolize the nodes of the graph corresponding to the warehouse.



Next is shown the robot with various sensors and add-ons.

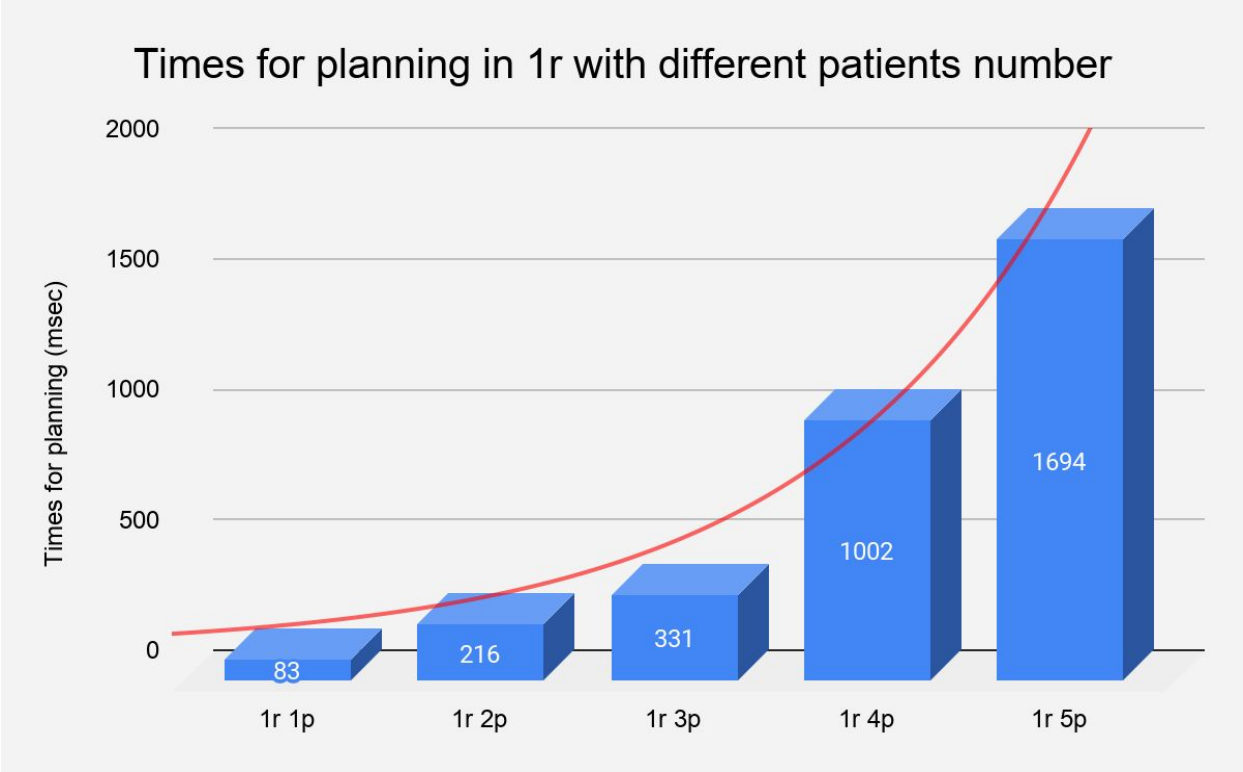
Section 5. Results and evaluation

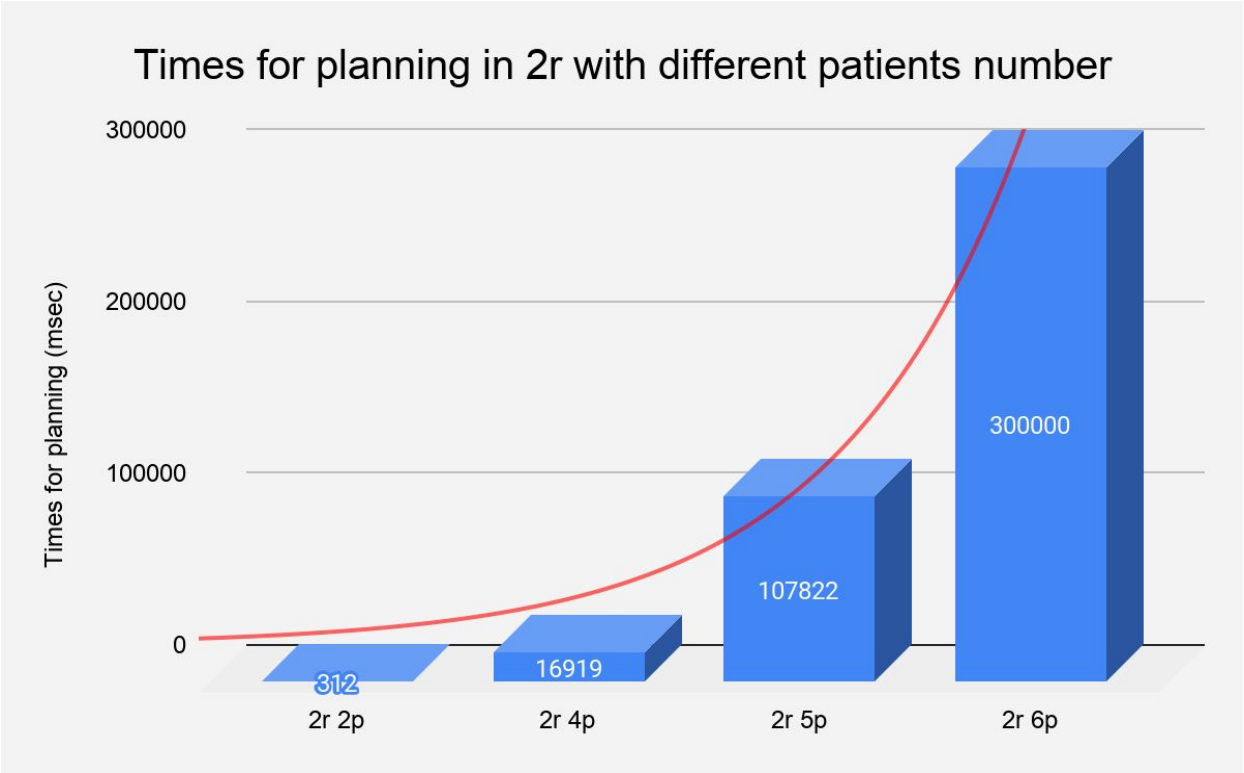
After developing a model for the planner, we did some empirical analysis of different instances of our problem, to evaluate the amount of time the planner spent to calculate the solutions.

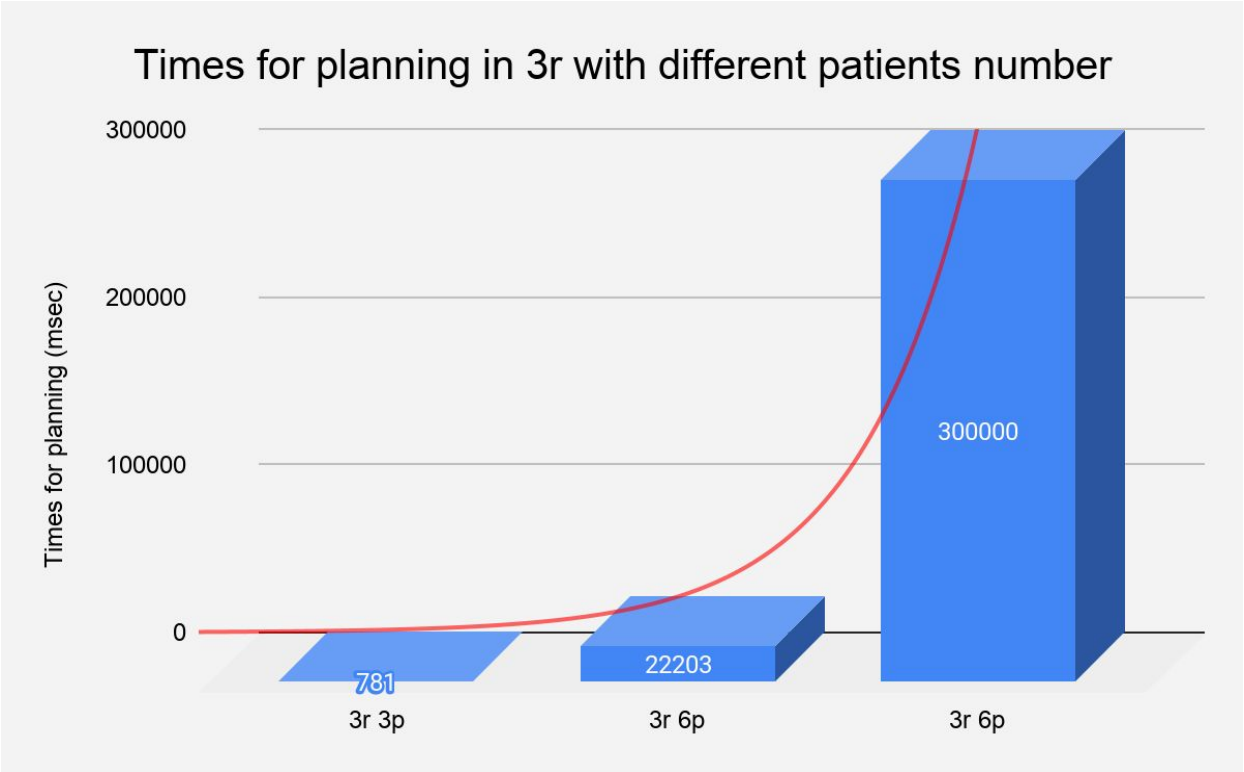
Given the nature of the constraints, we decided to proceed by requesting the model on the following parameters:

- number of rooms to visit
- number of patients per room
- number of times a room is visited within the same plan

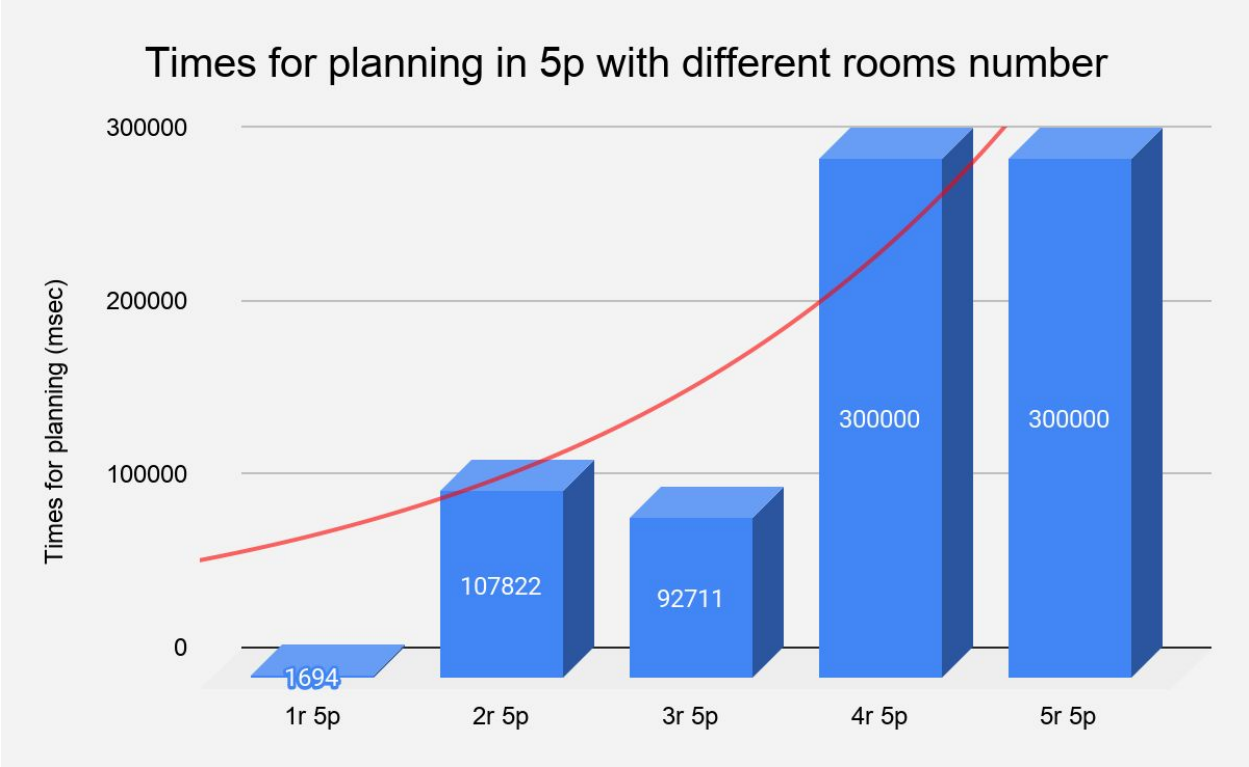
Below are the graphs obtained by changing the number of patients visited within one, two or three rooms:



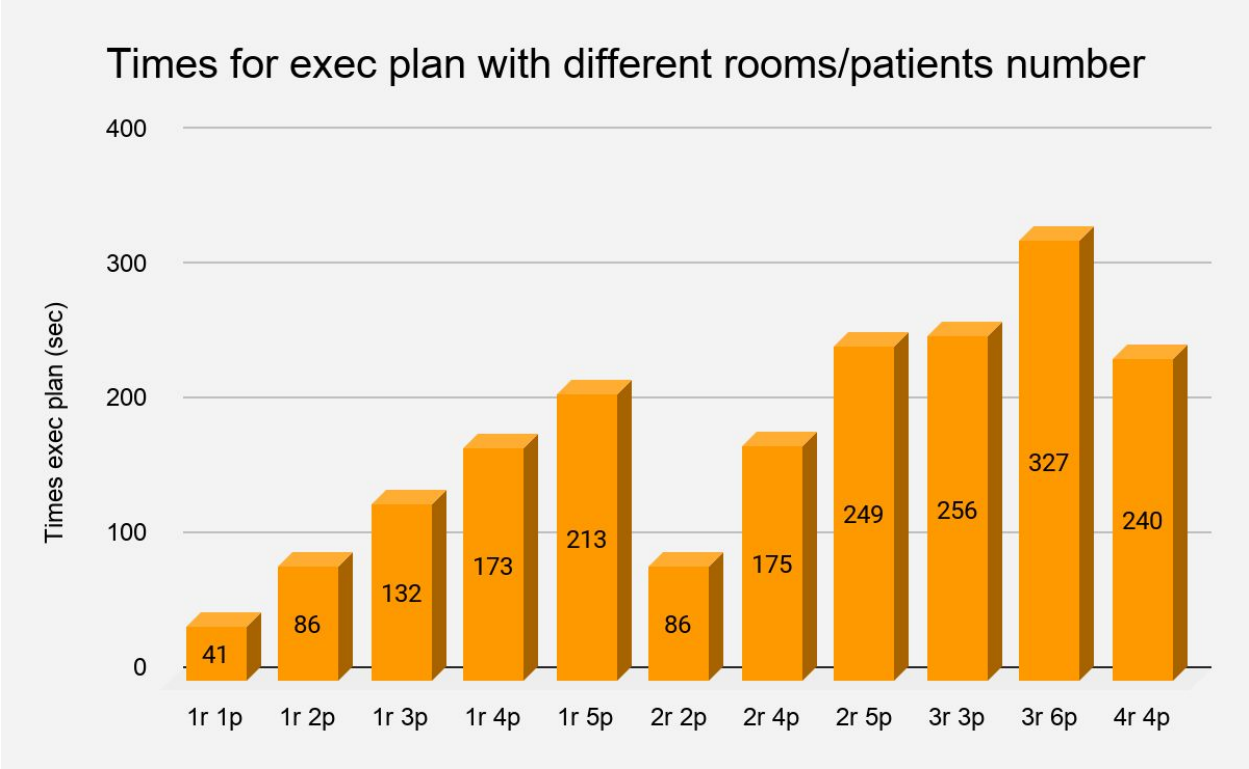

Times for planning in 2r with different patients number
Times for planning (msec)
300000
200000
100000
0
312
16919
107822
300000
2r 2p
2r 4p
2r 5p
2r 6p


Times for planning in 3r with different patients number
Times for planning (msec)
300000
200000
100000
0
781
22203
300000
3r 3p
3r 6p
3r 6p

The results show that the complexity of the model is exponential and explodes starting with instances of problems with only 6 goals. We also noted that due to the nature of the constraints in our domain, delivering drugs to patients in the same room makes the computation by the planner more complicated than delivering the same number of drugs to patients in different rooms.



A final analysis that was carried out, was to measure the times when the robot is in action for the achievement of goals during an entire shift of work.
It is to be noted that, also thanks to the fairly large ranges of input data in our . pdl, the robot respects all the time constraints of delivery of drugs. Another important aspect was the implementation of a system that suggests the robot to go inside a safe place during dead times. In this way the robot will not be in the way of the health staff; it will also be placed in a central position with respect to our hospital department which will allow it to reach more easily the next goals.
At the end of all this then it would be interesting to see how to refine the model for planning in order to allow delivery in a shorter time and with a much higher number of goals trying to contain the computational complexity.
Another interesting aspect would be the possibility of integrating the output of Platinum with the ROS system as Platinum allows both the planning and implementation of the plans generated through ROS.



Section 6. Conclusion and Discussion

One of the main problems that we encountered during the development of the project dealt with the management of the coordinates by Webots. Due to poor documentation, the correct use of components such as gps and compass proved more difficult than expected. To further complicate the situation, we have noticed that the examples offered are not always useful, as they are not very general.

Another problem is modeling and importing objects into Webots. Despite the wide range of objects made available by the simulator, being able to import new ones turned out to be a complex task. In order to recreate a bed as similar as possible to those actually found in hospitals, it was necessary to use a Webots add-on [7] for the graphic modeling engine, open source, Blender.
After being able to find a model of hospital bed that met our needs, we had to export it through the add-on in format .wbt and then converted manually to .proto, adding all the necessary properties.

This project has led us to look out over that branch of artificial intelligence that deals with generating and executing plans for machines independently with different degrees of freedom. It was interesting to learn how to model real world problems using state variables and time constraints and how this "declarative" approach leads to the solution of very complex problems only through their description.

It was also exciting to create a "real" controller for a truly existing robot used in different fields.

Finally, the technique of Pair programming (online) has been improved, as part of an agile development of the software. We tried to alternate as much as possible, based on the features to be developed, the driver and Observer roles using Git as a code versioning tool.
An unsolved problem, which we can place between future developments, is the integration between the avoidance collision algorithm (realized and already present and working within the project github repository) and the displacement between two points on the same segment.

A future development that could be made to the project is to simulate the "carriage" with a limited and variable capacity both from the point of view of planning and simulation, containing also different types of drugs so as to have a more realistic scenario.

Sitography

[1] Formula Logistica: https://www.formulaservizi.it/servizi/logistica/
[2] AUSL di Forlì: il processo di innovazione ed integrazione dei servizi ospedalieri di trasporto e di pulizia:
https://www.gsanews.it/terza-pagina/ausl-di-forli-il-processo-di-innovazione-ed-integrazione-dei-servizi-ospedalieri-di-trasporto-e-di-pulizia/
[3] Webots website: https://cyberbotics.com/
[4] PLATINUm GitHub repository: https://github.com/pstlab/PLATINUm
[5] Webots Pioneer 3-DX robot: https://cyberbotics.com/doc/guide/pioneer-3dx
[6] HospitalDrugsDelivery GitHub repository: https://github.com/Elidor00/HospitalDrugsDelivery
[7] Blender to Webots exporter add-on: https://github.com/cyberbotics/blender-webots-exporter
[8] Knowledge Engineering ENvironment for Timeline-based Planning (DDL.3 language)
https://ugilio.github.io/keen/index.html

Bibliography

[9] Cialdea Mayer, M., Orlandini, A. & Umbrico, A. Planning and execution with flexible timelines: a formal account. Acta Informatica 53, 649–680 (2016). https://doi.org/10.1007/s00236-015-0252-z
[10] Umbrico A., Cesta A., Cialdea Mayer M., Orlandini A. (2017) PLATINUm: A New Framework for Planning and Acting. In: Esposito F., Basili R., Ferilli S., Lisi F. (eds) AI*IA 2017 Advances in Artificial Intelligence. AI*IA 2017. Lecture Notes in Computer Science, vol 10640. Springer, Cham
[11] Umbrico, A., Cesta, A., Mayer, M. C., & Orlandini, A. (2018, June). Integrating resource management and timeline-based planning. In *Twenty-Eighth International Conference on Automated Planning and Scheduling*.
[12] Orlandini, A., Finzi, A., Cesta, A., & Fratini, S. (2011, October). TGA-based controllers for flexible plan execution. In *Annual Conference on Artificial Intelligence* (pp. 233-245). Springer, Berlin, Heidelberg.