ROBOCUP JUNIOR 2023 - RESCUE SIMULATION

ENGINEERING JOURNAL

TΔLΘS

Abstract

This engineering journal presents dated entries to show our progress throughout 2023. Although the team was formed in 2022, all previous documentation was only stored in our GitHub repository:

https://github.com/iita-robotica/rescate_laberinto/tree/master

A. Team

As of 2023, team Talos is formed by Alejandro de Ugarriza and Ian Dib. We started working together since March and coordinated all our work at the “Instituto de Innovación y Tecnología Aplicada” (IITA).

This was our work division in the beginning:

- Alejandro searched for new strategies regarding navigation, detection, and mapping.
- Ian tested different program alternatives to see which one performed better.

To share our respective progress, we held four-hour-long meetings each weekend at the IITA institute. Each of us worked separately during the week and communicated through discord whenever it was necessary.

Although we first considered rewriting our code in C++ to boost performance, we later discarded the idea as it was too time-consuming. Still, this progress was left in the Journal to show our journey

B. Table of contents

The journal follows the following outlay:

- Date (dd/mm/yyyy)
- Author of the journal page
- Tasks done
- Issues & Solutions
- Plans
- Research links
- Commit number

We decided to include a Commit number section to reference the GitHub commit that corresponds to the changes and tasks described in that journal page. This allows for quick look up in our repository (link in the abstract) in case the reader is interested in seeing the actual code presented.

Entries

This first part goes from December 2022 to March 2023.

Entry #1

Progress in **C++**

Date

03/12/2022 (week)

Authors

Alejandro de Ugarriza

Tasks done

- Updated Erebus (see [1]).
- Created the “Top Down” plan: divided the code’s structure in blocks.

Issues & Solutions

Issues

- Webots version 2021b doesn’t run in Ubuntu version 22.04.1.

Solutions

- Reinstalled Ubuntu version 20.04.1 (see [2]).

Plans

- Backup the program used in the RoboCup 2022.

Research links

[1] https://gitlab.com/rcj-rescue-tc/erebus/erebus/-/releases/v21.2.4

[2] https://davesroboshack.com/installing-linux/installing-ubuntu-as-dualboot/

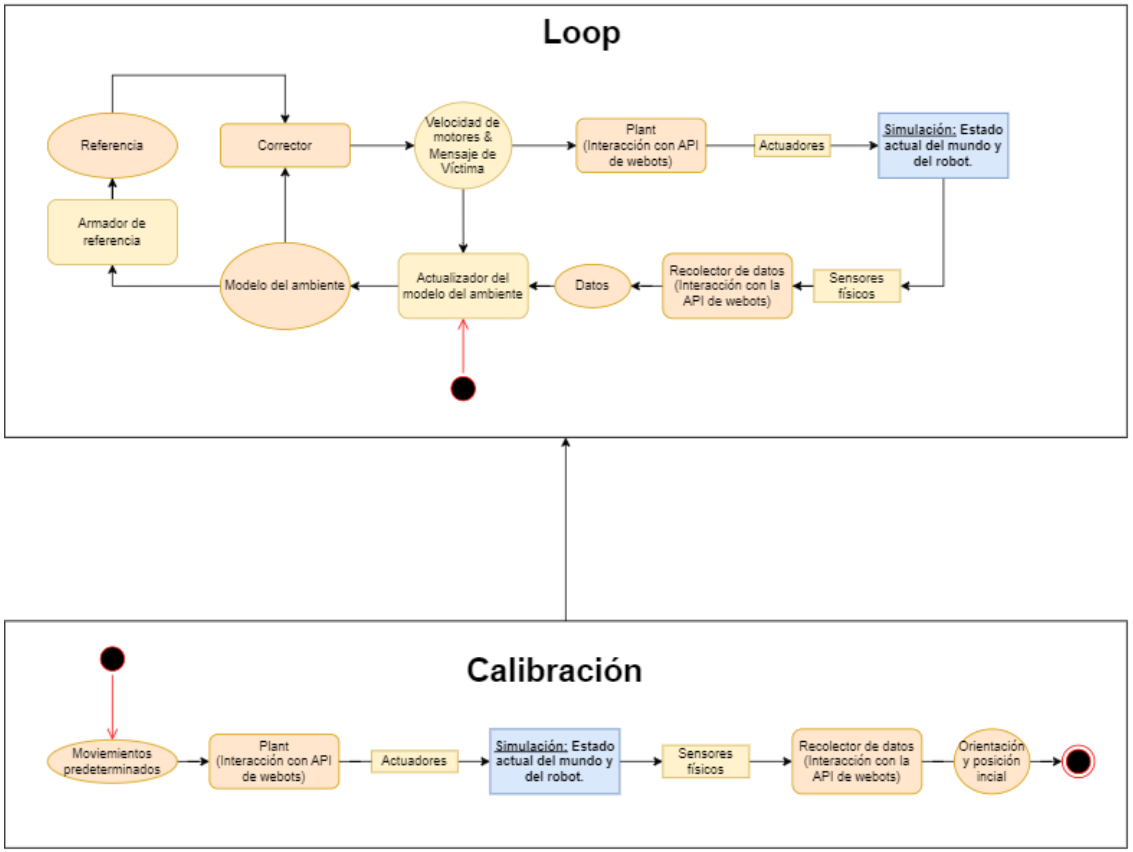
Commit number

-

Additional

- Designed a control diagram for the program (see Fig. 1). To reuse the diagram head to:

https://app.diagrams.net/#G1YOw1dpmXoxYFmws5oVhhKEpFKkhgYjnD



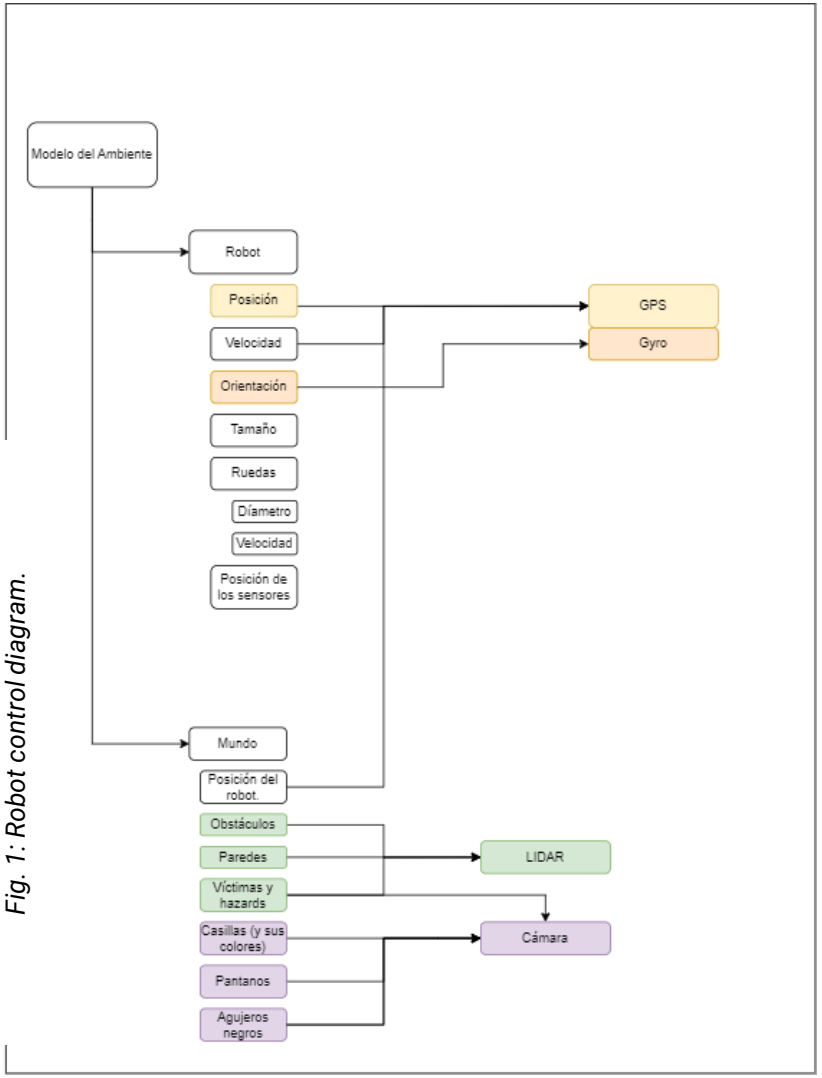


Fig. 1: Robot control diagram.

Entry #2

Date

10/12/2022 (week)

Author

Alejandro de Ugarriza

Tasks done

- Made a backup of the 2022 RoboCup competition files.
- Compiled the old program and its scripts to manage them easily.

Issues & Solutions

Issues

- Compiling bugs due to C++ not recognizing the Webots libraries.

Solutions

- Used CMake (see [2]), as detailed in the Cyberbotics documentation (see [1]).

Plans

- Create a new branch in the GitHub repository to store this year's progress.

Research links

[1] https://cyberbotics.com/doc/guide/compiling-controllers-in-a-terminal#macos-and-linux

[2] https://cyberbotics.com/doc/guide/using-your-ide#cmake

Commit number

-

```
$ export WEBOTS_HOME=/usr/local/webots
$ make
$ make clean
$ make -f Makefile_java my_robot.class
$ make my_robot.o
```

Fig. 2: Code to compile controllers in Linux.

Entry #3

Date

17/12/2022 (week)

Author

Alejandro de Ugarriza

Tasks done

- Created the new branch in the GitHub repository (see [1]).
- Researched the possibility of making a Mapping grid using series of bits (see [2]).

Issues & Solutions

Issues

- Unable to open .h files (header files).

Solutions

- Looked up for the correct syntax. Declared the external functions before calling them.

Plans

- Start writing simple blocks in C++, get the general functions working.

Research links

[1] https://github.com/CoolRobotsAndStuff/simulated_rescue_maze

[2] https://en.wikipedia.org/wiki/Bitmap

Commit numbers

-

Own elaboration adapted from https://hackingcpp.com/cpp/std/vector_intro.html

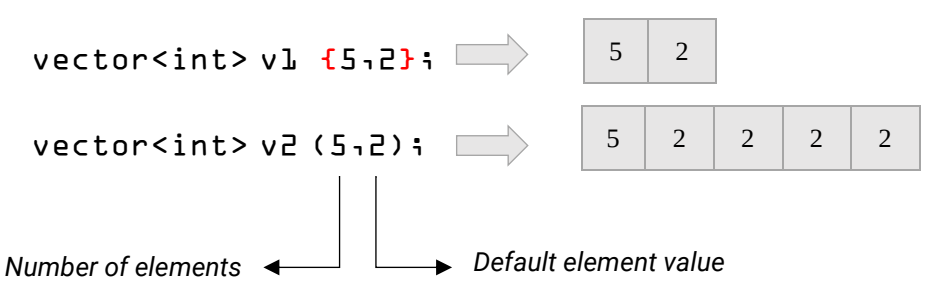


Fig. 3: C++ vectors

Entry #4

Date

24/12/2022 (week)

Author

Alejandro de Ugarriza

Tasks done

General progress:

- Created a script to generate makefiles with CMake (see [I]).
- Created a script to export the environment variable EREBUS_DIRECTORY. Added it to the .gitignore as it is specific to our system (see [II]).

Specific progress:

- Created the GPS, Gyro & Angles classes to work more easily (see [III]).
- Created the GPS block and a preliminary (but functional) Gyro block in the code (see [IV])
- Created the utilities & functions file to normalize angles and convert degrees to radians (and vice versa). Used the cmath library (see [V]).

Issues & Solutions

Issues

- The “cout” variable wasn’t printed in the Webots terminal.
- The gyroscope may not be precise enough. More testing needed to reach conclusions.
- Saving angles as doubles makes them more tedious to normalize and convert.
- The Angles class doesn’t have a great interface yet. Need to learn the skills to improve it.

Solutions

- Added a new line (“endl”) at the end. Changed the syntax from “cout << x” to “cout << x << endl”.

  The other issues still need to be investigated.

Plans

- Search for an angles’ library (or keep working on ours).
- Sensors classes will be named after the sensors itself, separating them from the Webots classes by using a different namespace.

Research links

-

Commit numbers

[I] d48b3999dc77bee8b9d473cb1a40775d14622ed5

[II] 5216a06a319c37b2eef4c41869ca9bcbd1b147e1

[III] 06222759c83be310562a17346eaf4208a39012a0

[IV] 2a4dbc4ac23f7804060cade5ca005136c6d3e3df

[V] B06d60328cdc612e07aa1fe9a1d819a9ace67281

Own elaboration adapted from https://preshing.com/20170511/how-to-build-a-cmake-based-project/

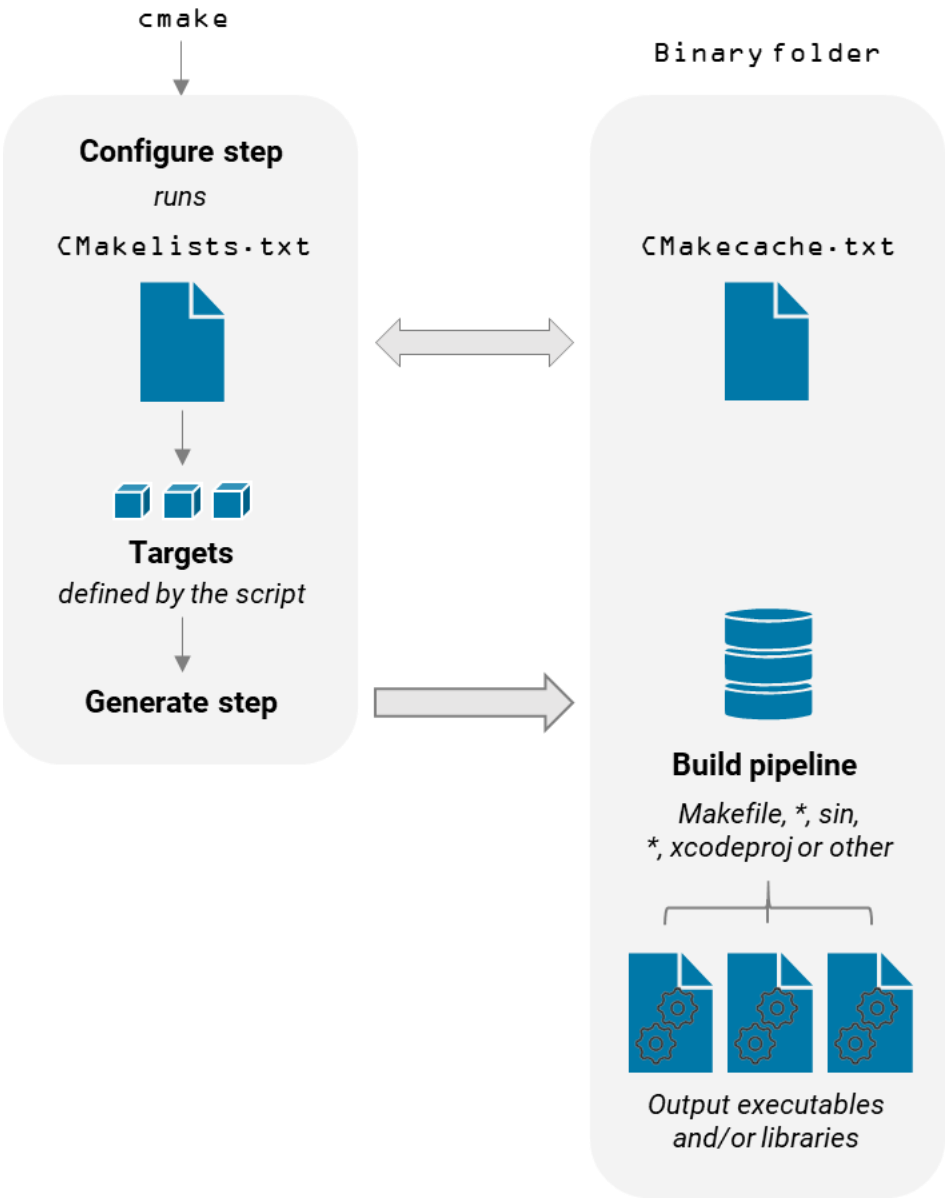


Fig. 4: CMake-based project

Entry #5

Date

01/01/2023 (week)

Author

Alejandro de Ugarriza

Tasks done

- Created the Sensors & Actuators classes (see [I]).
- Created the FloatVector3D & DoubleVector3D classes (see [II] & [II']).
- Created the Robot, Camera, Wheels & LiDAR classes to save the robot's model. Still need work (see [III]).
- Adapted the GPS & Gyro classes to use the DoubleVector3D instead of structs to save data (see [IV])
- Changed the headers extensions from .h to .hpp (see [V]).

Issues & Solutions

Issues

- The .gitignore can't detect the compiled files. An extension is needed.

Solutions

- Modified the compile.sh script to name the compiled files "code.out" instead of "code" (see [VI]).

Plans

- Need to establish rules to create the namespaces. Otherwise, it gets confusing.

Research links

-

Commit numbers

[I] c648f0f6272eb946f92a2bee8b87ee78c015fcd8

[II] 304a884c6701554d4909d6e53619ef5fdc7f10f7

[II'] a0683b37ebc3442fb8f0cb0ba07a7b7fa78414fc

[III] 32cc15dd4fdb5ede345c19c50816b2626baa31ec

[IV] e20982807491b7169a2d29340ebfcab8645bda97

[V] f7798e90ee388a46357ecf02dbe1b46a38e8047b

[VI] d992492b94476d8d5e1d54f18a8f3b662b0f23d9

Entry #6

Date

08/01/2023 (week)

Author

Alejandro de Ugarriza

Tasks done

- Created the SensorManager class to organize sensors (see [I]).
- Created the SensorDataLoader class to upload the sensors' data (SensorManager) to the robot (see [II]).

  Added the init & default constructor classes to the GPS & Gyro sensors.
- Added a print method to the FloatVector3D and DoubleVector3D classes (see [III]).
- Modified the position, velocity, and acceleration in the environment model of the DoubleVector3D to make it compatible with the sensors' data (see [IV]).
- Modified the sensors' blocks to print their value using the FloatVector3D class (see [V]).

Issues & Solutions

Issues

- Redefining errors in files that were imported more than one time (see [1]).

Solutions

- Added "#pragma once" to avoid redefining errors (see [I]).

Plans

- Continue developing the algorithm angle & position corrector in the robot.

Research links

[1] https://stackoverflow.com/a/34873122

Commit numbers

[I] 15e92515e3053e568a34981da1045fe3cfd02c1c

[II] 4d441f1eed4e3cc9a9c410471667974596cc0c57

[III] ebd10f92ee4b6b24fe53cdf672a49c1ab84a3a50

[IV] 4abba88bbdfc1c0f4fdce26ef977f0727c604e13

[V] e7fb6386f858799b4a029de1a2692adb48f969e4

Entry #7

Date

15/01/2023 (week)

Author

Alejandro de Ugarriza

Tasks done

- Created the DifferentialVelocities class in generic_data_structures to save the wheels velocity (see [I]).
- Created the AngleRotator class to control the robot's turn and make them precise (see [II]).
- Created the CoordinateMover class to specify the coordinates the robot should move to.

Issues & Solutions

Issues

- Can't create a generic function for doubles, floats and ints. Need more research.

Solutions

- No apparent solution in C++. Instead, a different approach was taken:
- Created the change_range function to work with doubles & floats (see [III]), similar to Arduino maps (see [1]).

Research links

[1] https://reference.arduino.cc/reference/en/language/functions/math/map/

Commit numbers

[I] bbdfe06638f3cd5e221aa34c27c6d76fb4d2543e

[II] be78973c16c936b3a8dafe173d58459cae56396c

[III] c4e8ab4ea0df72f57bc3b69865660aa1e5da1384

Own elaboration adapted from https://blog.feabhas.com/2021/07/cmake-part-2-release-and-debug-builds/

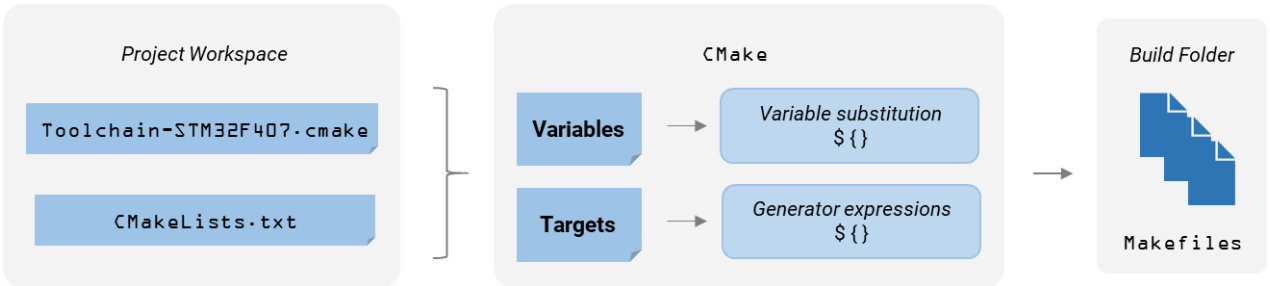


Fig. 5: CMake generator expressions

This second part goes from March 2023 to June 2023. Ian joined team Talos to test different versions of the program and document the process. In March, he studied the program's advances up to that point.

Entry #8

Date

06/03/2023 (week)

Progress in **Python**

Author

Alejandro de Ugarriza

Tasks done

Overall optimization of the 2022 version of the code using g-profiler. Other changes:

- Optimized the images processing functions to use more capabilities of the NumPy library.
- Modified the grid's function to return a reference instead of creating a copy.

Issues & Solutions

Issues

- NumPy no longer accepts np.int; it was cut to just int (see [I]).

Solutions

- Checked the NumPy documentation and stackoverflow (see [5]). Changed the code's syntax.

Plans

- Check for other improvements in the code.

Research links

[1] https://python.plainenglish.io/how-to-identify-bottlenecks-in-your-python-application-5f3d680a96c0

[2] https://towardsdatascience.com/how-to-find-out-the-bottleneck-of-my-python-code-46383d8ef9f

[3] https://python.plainenglish.io/how-to-identify-bottlenecks-in-your-python-application-5f3d680a96c0

[4] https://stackoverflow.com/questions/45893768/how-do-i-find-out-what-parts-of-my-code-are-inefficient-in-python

[5] https://stackoverflow.com/questions/48085507/converting-a-numpy-array-to-integer-with-astypeint-does-not-work-on-python-3-6

Commit numbers

[I] 7ba96e0fa14ffa17a19cff7b39e8d2186b26fed9

Entry #9

Date

13/03/2023 (week)

Author

Alejandro de Ugarriza

Tasks done

- Reconstructed the .json file of the robot's configuration so that it worked in Erebus v23.0.0+ (see [I]).
- Added flags to activate or deactivate the debug information (see [II]).
- Increased victim detection wait time (see [III]).

Issues & Solutions

Issues

- The getData receiver function no longer works.
- The LiDAR scanner doesn't get proper detections.

Solutions

- Modified the getData function to the getBytes one (see [IV]).
- Modified the LiDAR scanner syntax (see [V]). Still needs to be adjusted.

Plans

- Check for other improvements on the code.

Research links

-

Commit numbers

[I] 80502cf4bc47db7a96529733eeb1f684a7113306

[II] 4fb04ee39d123db53e8f0059a68c0f876e0a777a

[III] b09c1c6d96828ac730738b534739a4bd77e5992f

[IV] 9d00aaf836dcebe672a276192e22210e03530165

[V] 4d8368b393914e8933b0ca02522d385eff0dd2c4

Additional

- Configuration of the robot's sensors comparison (see Fig. 6).

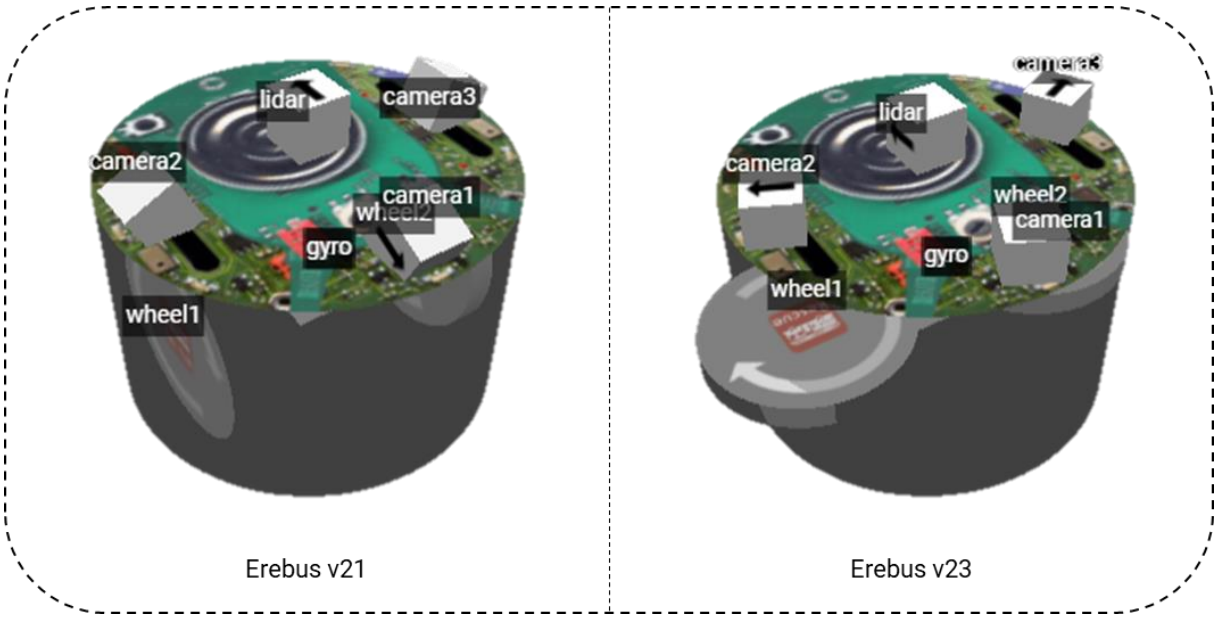


Fig. 6: Comparison of the previous .json robot's configuration file in both versions of the customization tool (the coordinates used in the customization tool seem to have changed)

```
SHOW_FIXTURE_DEBUG = False
SHOW_DEBUG = False

SHOW_GRANULAR_NAVIGATION_GRID = True
SHOW_PATHFINDING_DEBUG = False
SHOW_BEST_POSITION_FINDER_DEBUG = False

PRINT_MAP_AT_END = True
DO_WAIT_KEY = True
```

Fig. 7: Flags.py file for debugging

Entry #10

Date

20/03/2023 (week)

Author

Alejandro de Ugarriza

Tasks done

- Created the Angle class. Moved the functions related to that datatype from the utilities.py file to the class.
- Created the Sensor, timedSensor & StepCounter classes, which allow to execute actions every n number of steps (see [I] & [I'']).

  Example: the functions that process camera data can use the StepCounter to only repeat the process every three steps, saving computation time.

  *This was previously implemented as a decorator, but we decided it was better represented as a class.

Issues & Solutions

Issues

- Victim detection and flag for fixture detection debugging no longer work.
- One camera is turned 180°, despite its rotation on the .json file.

Solutions

- Modified the detection functions and the flags file to match the new syntax (see [II]).
- Modified the image's rotation on the camera software, instead of on the customization tool (see [III]).

Plans

-

Research links

-

Commit numbers

[I] a814335545e57e1dcb8273d87442d3b798616878

[I'] 538bb194c325e5e7efaa17db764376b07bcbee93

[II] ad9720b562c64ba182e464ea497831671f8d772c

[III] b580dfb5dc14e490d84ea301547c39cfc79a48f4

Entry #11

Date

27/03/2023 (week)

Author

Alejandro de Ugarriza

Tasks done

- Created classes to represent 2D-positions and 2D-vectors. Moved frequently used functions such as get_distance_to() or get_angle_to() to these classes (see [I]).
- Changed position related variables in the robot.py, main.py and devices/gps.py files to use the Position2D class instead of lists (see [II]).
- Commented the code. Added more global debug flags to control the information that is shown more precisely (see [III]).

Issues & Solutions

Issues

- The robot class is not working properly.

Solutions

- Encapsulated low level movement (precise rotation and movement) in the classes DriveBase, RotationManager and MovementToCoordinatesManager, inside the low_level_movement directory (see [IV] & [V]).

Plans

-

Research links

-

Commit numbers

[I] 98809e46954fd7b9cee8619131e1f090afaa4781

[II] 0491084cbaeb1abee9f8e1dd7194aedc6763d2d8

[III] 0bf4a53d773f8c6ebd6e930665a099c864b22df4

[IV] 00c32c1ff7d2669eb7a8d549f9656c2f3420b706

[V] 162f00b49d1c7f5ba7b86391006974623e7b8444

Entry #12

Date

03/04/2023 (week)

Author

Alejandro de Ugarriza, Ian Dib

Tasks done

- Created the Fixture_Detection class. Moved all related variables to the class (see [I]).

  *This was previously a collection of global functions and variables, not very organized.

- Created the filter_values class to change the values and priorities of each color filter (see [II]).

  *There used to be a function for each victim, which slowed down the program.

- Improved victim classification. The categorization is now achieved by dividing the center of the image vertically into three zones and counting the black pixels in each (see [III]).

Issues & Solutions

Issues

- Some unnecessary files were left in the code after the previous commits.
- Categorization of victims cannot be precisely achieved when the camera catches only half of the image.

Solutions

- Deleted the timing_constants.py file (previously used to set functions to execute every n timesteps). All values are now stored inside the StepCounter instances (see [IV]).
- Modified the division of the three vertical zones. Instead of making the division in the image's center, the agent now does so in the image's centroid, reversed in its x axis (see [V]).
- As the divisions are now made in the part of the image with the least black pixels, the new detection system has shown a better performance at categorizing incomplete images than other systems such as OCR.

Commit numbers

[I] 28a4fed794d94d7af134559c740aa57afc92de33

[II] 4028f9de14c17562a257879467a4aec032195021

[III] 3fb540a000e8b44eed48c9d6564b4802222da6ea

[IV] aad136d904994f13b22b7cedd64d9ddd0617caad

[V] 0d22324c9c99ad4f9440420b8a8b9755f103f812

Own elaboration adapted from Mustafa (2013) with permission:
https://www.researchgate.net/publication/260405352_OPTICAL_CHARACTER_RECOGNITION_OCR_SYSTEM_FOR_MULTIFONT_ENGLISH_TEXTS_USING_DCT_WAVELET_TRANSFORM/figures?lo=1

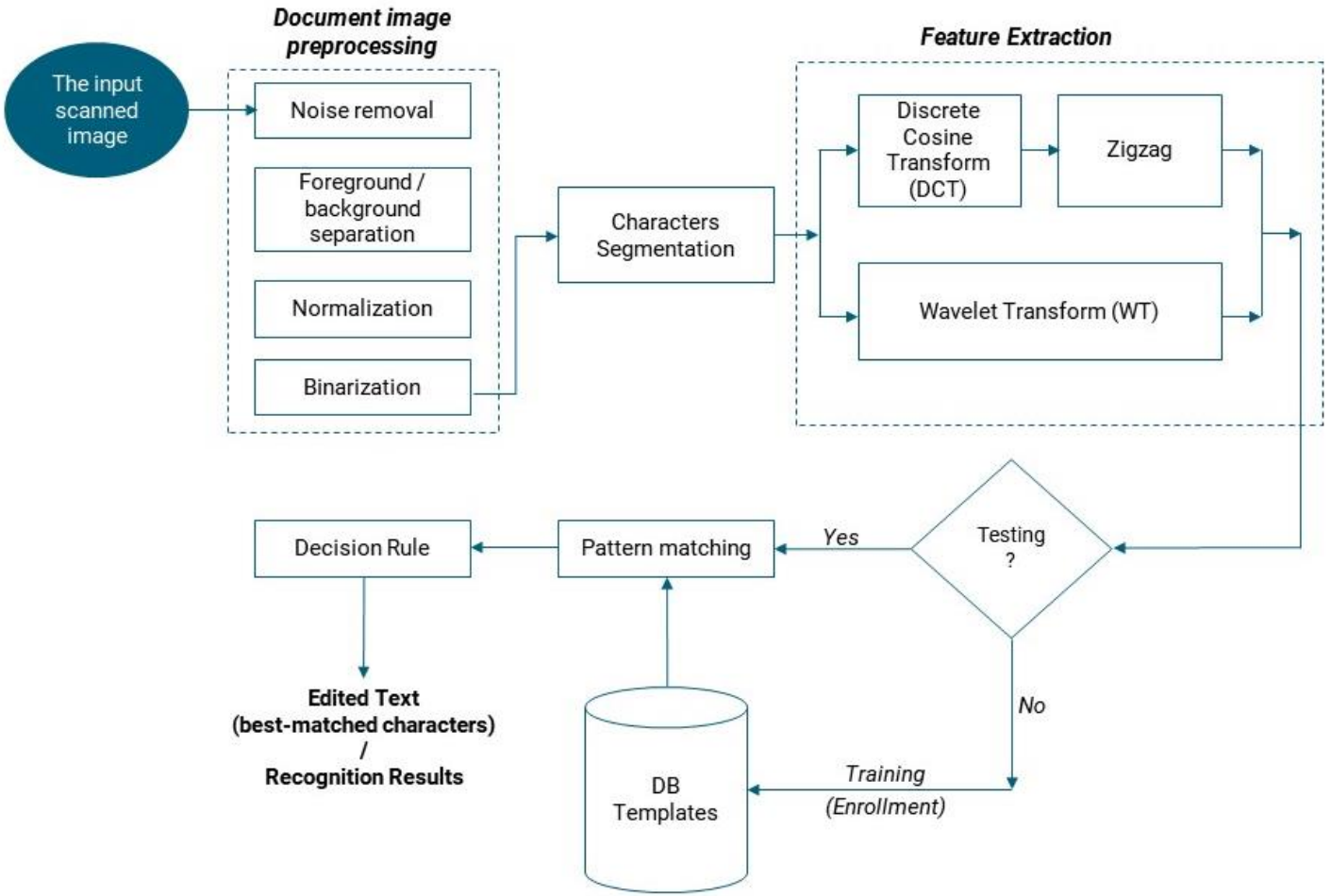


Fig. 8: OCR detection diagram

Entry #13

Date

10/04/2023 (week)

Authors

Alejandro de Ugarriza, Ian Dib

Tasks done

- Copied the latest code version to the src/ directory for easier development (see [I]).
- Created scripts/ directory and copied the script to change the import path of the run.py file there (see [II]).
- Created robot_jsons directory and copied json file of robot there (see [III]).
- Moved old folders to a new branch, working as an archive of previous code iterations (see [IV], [V] & [VI]).

Issues & Solutions

Issues

- As there used to be a branch for each year's advances in the code, re-writing sections of the code often involved fixing errors regarding missing files.

Solutions

- We moved the code to the src/ directory. Each year's progress is marked with a "release".

Plans

-

Research links

-

Commit numbers

[I] 6adc0e7559311897a64727b4c389912c2c666ad6

[II] cc75180937d02ad21a0ca0b2146912e792cf0c74

[III] 7a8e5cda64d656094bd1c08cc4a48b3a97eddc54

[IV] 671ffe912aef8c8564e1d52995dcf0b4487289b1

[V] 751cc36d537bd75e1687d6a718ecce2450ece32f

[VI] 223bd3e559815dba4dc2234363d93320b38b0e39

Entry #14

Date

17/04/2023 (week)

Authors

Alejandro de Ugarriza, Ian Dib

Tasks done

- Developed a resizable grid to represent the points cloud obtained by the LiDAR. Used the filter2d OpenCV function to calculate the margins of each point in the points cloud (see [I]).

  *As this function is normally used to apply effects to images, it is highly optimized.

- Added the Grid class to Mapper and added a flag to the flags.py file to print it (see Fig. 10) (see [II]).
- Added the debug option for the Grid to the flags.py file (see [III]).

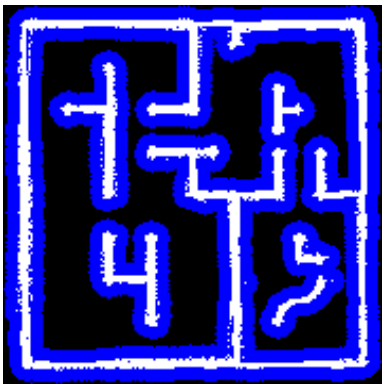


Fig. 10: Resizable grid

Issues & Solutions

Issues

- The margin given to each point in the points cloud is not precise enough for proper navigation.

Solutions

- Applied a circular kernel to the OpenCV function to obtain better results.

Plans

- Ignore the concept of “tile” to design a more general algorithm.

  *Particularly useful in area 4, but overall an improvement in all areas.

Research links

https://www.askpython.com/python-modules/opencv-filter2d

Commit numbers

[I] 0370a95d7b0f9bc3b9246b0598288793b56e3dcd

[II] 44da52f0b361fbbdb458a523242d34bcb1306b4d

[III] 1e2e032135da53d7ce6e766bbb5f3f451a5ed0fc

Entry #15

Date

24/04/2023 (week)

Authors

Alejandro de Ugarriza, Ian Dib

Tasks done

- Added to_int() and get_np_array() methods to Position2D class (see [I]).
- Used Position2D for all pertinent variables in Mapper class. Changed order of functions in Mapper for easier navigation. Made some methods private (see [II]).
- Changed victim reporting method names in Mapper to be more explicit and minimize confusion (see [III]).
- Added methods to convert from GPS coordinates to indexes –traditional tile-based navigation data structure– (& vice versa) for the navigation grid (see [IV]) and a function to get the closest quarter-tile vortex for the Mapper class (see [V]).

Issues & Solutions

Issues

- Navigation was slow and meeting some difficulties.

Solutions

- We changed the order of functions in Mapper.

Plans

- Implement A* algorithm to the navigation, making use of the new grid.

Research links

-

Commit numbers

[I] e8293eb53cd0cb373af05f296c884954526056b8

[II] ff437093aedaab80ffa9eb3bf7aca797b63619ca

[III] 8980db4debb9589de86f4c8a9b0d31408dcdd9a6

[III'] 21f6b22a55982d31f5e59efe34c79e5ecb781712

[IV] 330024049667a794faaa0a0879d79381875df4ec

[V] eff09fb7c2cdc1aaf6bbac7282083cb49fd2bf64

Entry #17

Date

01/05/2023 (week)

Authors

Alejandro de Ugarriza, Ian Dib

Tasks done

- Modified the Agent class to accommodate to the granular grid (see [I]).
- Developed a Breadth-first search algorithm to find the closest available position in case the current robot position appears to be not traversable (see [II]).
- Developed a working A* algorithm for navigation (see [II]).
- Modified the grid to expand dynamically in any direction, so it can take negative indexes (see [III]).

  *Still need to access the NumPy arrays directly (to avoid crashing the program), so we use methods to convert between the array_index and grid_index.

Issues & Solutions

Issues

- The robot works slowly when moving as the A* algorithm indicates.
- The robot ends up in positions mapped as “not traversable” and can’t get out.

Solutions

- No solution yet.
- Reduced error margins in low-level movement and made it more precise (see [IV]).

Plans

-

Research links

-

Commit numbers

[I] 6971eda925d7f7b4aa9366e6cdca5d6333a1a756

[II] 64d28e5052d0216c716f097332754d8413a16520

[III] 12fd5f27b92852a90591a7884a649dec045b441b

[IV] c432666ebf9b8be227f6802f6315f35700da3418

Entry #18

Date

08/05/2023 (week)

Authors

Alejandro de Ugarriza, Ian Dib

Tasks done

- Developed a new array into the navigation grid representing the “preference” of a certain point for navigation i.e. points farther away from the wall are preferred for navigation (see [I]).

  *A green gradient indicates preference (stronger color == less preferred)

- Modified the A* algorithm based on a stack overflow response (see [1]). Implemented a lookup table for the best g score of a point, instead of looping trough all points and seeing if there was a better one (see [II]).
- Modified the granular grid to reduce the “robot margin” (blue color) to the minimum possible extent, to gain more precision (see [III]).

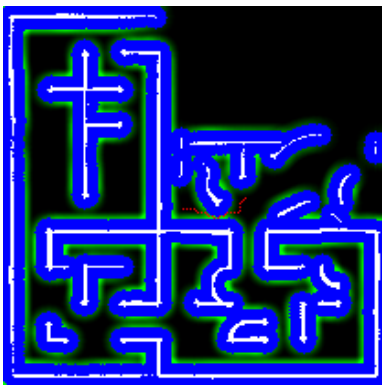


Fig. 11: New array

Issues & Solutions

Issues

- When rotating to an angle, the robot “vibrates” back and forth, causing erroneous orientation measurements.
- When expanding the navigation grid, some points of the points cloud were placed in an “offset” position.

Solutions

- Reduced the precision of each rotation depending on the angle (see [IV]).
- Fixed wrong indexes: x[0] should have been x[1] (& vice versa) (see [V]).

Research links

https://stackoverflow.com/a/62074332

Commit numbers

[I] 8feceba15f4352f76ce04cac9637d55fbdb487dc

[II] e577b3aa8e3134523bace615cd17c9f0353d05df

[III] 5d0cb31fe6048ecce4bf34f069d484d79e2a7e9a

[IV] 7f0042295996968fea707c472c6a184d59de7f38

[V] c8d7c5b75ce602106b1e2e24974458ec8b188b51

Entry #19

Date

15/05/2023 (week)

Authors

Alejandro de Ugarriza, Ian Dib

Tasks done

- Improved navigation: reduced the weight of the preference grid in AStar from 50 to 5. This drastically improves performance in areas with narrow passages (see [I]).
- Improved mapping: reduced robot radius to the bare minimum, so it accurately detects narrow but possible ways (see [II]).
- Added method to generate a linear circle template for the preference matrix as an alternative to the current exponential one. This ended up not being used (see [III]).
- Developed smooth low-level movement: our technique relays upon the angle difference to calculate direction and speed, and on the distance to the target position to calculate the rotation speed (see [IV]).
- Improved the A* algorithm: when calculating the path, takes the average of the current, previous and next points in the path (see [V] & [VI]).
- Improved movement: when the difference between two angles is greater than 45°, the robot can turn on its own axis to prevent crashing against the walls (see [VII]).

Issues & Solutions

Issues

- When the node where the robot stands is occupied, the check path function erroneously returns True.

Solutions

- Removed unnecessary checks from the check path (see [VIII]).

Commit numbers

[I] fba03a91a4adaebdd3a56eb1134d0b966a3b1142

[II] f09d700955ce5c72dea2523c167393d79d13507b

[III] 0d56252fb230b0906606c4ac2273f9686283e300

[IV] 05ef087f64081cbddf67854a6526e30d96cca3bf

[V] daf0fa834de9ea3ec474c138527615d71f953bf4

[VI] a2eb0dd43bd7a31d31550c22b97e3c82ba9f53e8

[VII] 577d8a6b096a6b4750d7206a28bc1cc0ff39b0c8

[VIII] 3fb05e7ffa7ae991eedf3e013e2e0f7f0d8d6067

Entry #20

Date

22/05/2023 (week)

Authors

Alejandro de Ugarriza, Ian Dib

Tasks done

- Improved the grid expansion: moved all arrays of compound pixel grid into a dictionary to ease batch operations.

  *The arrays of compound pixels used to be individual variables.

- Created an array for the points already traversed by the robot (see [II]).

  *The purple gradient indicates the areas traversed by the robot.

Fig. 12: New array

Issues & Solutions

Issues

-

Solutions

-

Plans

- Continue with the calculation of the best point to explore:

  1. Create an array that shows the points seen by the camera: create a template with the cameras vision.
  2. For each point draw a line of pixels to the position of the robot.
  3. Check whether it is valid.

Research links

-

Commit numbers

[I] da900ef32f0337f9c22223ae92e76599eecbe98f

This third part goes from June to the competition itself in July. As such, it is still a work-in-progress, mainly filled with plans and strategies to tackle the tasks that are still left.

Entry #21

Date

29/05/2023 (week)

Authors

Alejandro de Ugarriza, Ian Dib

Tasks done

- Changed the way tiles work by pixels -> approx. 10 pixels x tile
- Improved victim detection -> After modifying the camera rate
- Changed the classification zone on the victim.
- A_star is working in the tile navigation using pixelated tiles. Still needs optimization and setup.
- Generation of a test environment to validate changes in each PR.
- Evaluation of the camera performance, adjusting position/angle for victim detection.
- Generation of maps where the robot passes through places where the margin between the robot size and the walls is small. See that the robot adjusts to the camera limitations instead of choosing the optimum path, so that no victim is left behind.
- Algorithm to detect the tiles color.
- Long-distance victim detection.
- Generation of final map with conversion of pixels to CSV.
- Algorithm to detect patterns and correct.

Issues & Solutions

Issues

- The Lidar image processing needs optimization.
- The diffdrive speed control couldn't be fixed at this point.
- Moving the robot from one point to the next, the need to adjust the camera became apparent.
- There was the need to look for the nearest wall not seen by the camera.
- Generation of virtual environment to detach ourselves from the host.
- Check data from the camera with Lidar on the grid.
- Code reorganization.
- Detection of tile color with new algorithm.

- Color detection had to be improved.
- Initial window in the virtual environment must be fixed.

Solutions

- When adjusting the camera to 90, the best image is obtained compared to other angles.
- Search for the farthest point that the camera can detect.
- A new algorithm was used for color detection.

Plans

- Navigation using pixelated tiles
- Diffdrive speed control.
- Behavior Tree -> Avoid sequential state machine
- Run a test to optimize/ verify that a_star works.
- Review margins, i.e., when the robot moves too near the wall.
- Optimize a dynamic margin in astar.
- Solve the problem when the robot is inside the forbidden margin.
- Modify Lidar threshold (current frequency is every 6 timesteps -> 6*32ms -> 5,2Hz).
- Add comments in code.
- See the possibility of disabling Lidar once the map has been recognized.
- Work on the next destination.
- Generate a new testing environment to validate the changes in every PR.
- Add comments to the code.
- Test the new grid.
- Avoid going through swamps with the astar to improve speed.

Entry #22

Date

05/06/2023 (week)

Authors

Alejandro de Ugarriza, Ian Dib

Tasks done

- Detection of far-away victims.
- Positioning in front of the victim.

- Victim classification.
- Use of a circumference to approach the victim and classify it
- Victim detection has been improved.
- Victim classification achieved.
- Map generation.
- Migration of changes to the testing repository.
- Fixing of controllers so that the mission can start automatically.
- Go over the map route to follow walls.
- Filter to detect false walls.

Issues & Solutions

Issues

- False walls must be filtered. These appear when there is an isolated pixel.
- There was an error when the victim detection message was generated.
- Need to generate the petition of time to end the game.
- The wheels move, but the robot doesn't.

Solutions

- The petition was generated but needs testing.
- A filter was implemented to detect false walls.
- The error in the victim detection message was fixed. The GPS crude data had not been used and was therefore incorporated.

Plans

- Test victim classification.
- Generate a complete map.
- Test lack of progress.
- Translate tile detection to a map structure.
- Compile more info to save in the CSV: time, m^2 traveled, number of victims detected, number of swamps, and a grid print screen.
- Finish the code to approach the victim using a circumference and a margin.
- Merge to a main branch ("New_navegation -> ale/dev"). Once this is done, update the solution to transmit the crude GPS data.

Entry #23

Date

12/06/2023 (week)

Authors

Alejandro de Ugarriza, Ian Dib

Tasks done

- Fix victim detection with current camera angles.
- New agent with sub-agents (victim, checkpoint, timeout).
- Add position to victims in the map.
- Change of illumination.
- Generation of agents' manager.
- Setup of each agent (wall follow, explorer, victim, checkpoint).
- Instructions to go to checkpoints.
- Optimization of short-term movements.
- Fixing of bugs.
- Testing of victim captures to validate classification.
- Calculation of victims detected.
- Testing in 18 maps.

Issues & Solutions

Issues

- Problem in victim detection due to a change of position in the camera.
- False victim detection.
- Problems in victim detection due to illumination issues.
- Problem in obstacle detection due to the camera.
- Threshold generates problems with black hole detection.
- Timeout needed to be fixed so that it didn't need to return to the beginning.

Solutions

- Change of code to detect victims with the new camera position.
- False victim detection was fixed with Lidar.
- Reduction of map errors.

- Victim detection due to the camera position was solved.
- The timeout issue was solved but is still disabled.

Plans

- Improve checkpoint detection.
- Astar: review which node is nearer to the robot (recalculation).

References

<https://gitlab.com/rcj-rescue-tc/erebus/erebus/-/releases/v21.2.4>

<https://davesroboshack.com/installing-linux/installing-ubuntu-as-dualboot/>

<https://app.diagrams.net/#G1YOw1dpmXoxYFmws5oVhhKEpFKkhgYjnD>

<https://cyberbotics.com/doc/guide/compiling-controllers-in-a-terminal#macos-and-linux>

<https://cyberbotics.com/doc/guide/using-your-ide#cmake>

<https://github.com/CoolRobotsAndStuff/simulated_rescue_maze>

<https://en.wikipedia.org/wiki/Bitmap>

<https://hackingcpp.com/cpp/std/vector_intro.html>

<https://preshing.com/20170511/how-to-build-a-cmake-based-project/>

<https://stackoverflow.com/a/34873122>

<https://blog.feabhas.com/2021/07/cmake-part-2-release-and-debug-builds/>

<https://python.plainenglish.io/how-to-identify-bottlenecks-in-your-python-application-5f3d680a96c0>

<https://towardsdatascience.com/how-to-find-out-the-bottleneck-of-my-python-code-46383d8ef9f>

<https://python.plainenglish.io/how-to-identify-bottlenecks-in-your-python-application-5f3d680a96c0>

<https://stackoverflow.com/questions/45893768/how-do-i-find-out-what-parts-of-my-code-are-inefficient-in-python>

<https://stackoverflow.com/questions/48085507/converting-a-numpy-array-to-integer-with-astypeint-does-not-work-on-python-3-6>

<https://reference.arduino.cc/reference/en/language/functions/math/map/>

<https://www.askpython.com/python-modules/opencv-filter2d>

<https://stackoverflow.com/a/62074332>

Al-Hassani, Dr. Mustafa. (2013). Optical Character Recognition (OCR) System for multifont English Texts using DCT & Wavelet Transform

<https://www.researchgate.net/publication/260405352_OPTICAL_CHARACTER_RECOGNITION_OCR_SYSTEM_FOR_MULTIFONT_ENGLISH_TEXTS_USING_DCT_WAVELET_TRANSFORM/figures?lo=1>