Hochschule Reutlingen
Fakultät Technik

Computer Science:

Documentation for the Tik Tak Toe Robot:

Name: Felix Schnoor (767503) / Julian Benjamin Maier (767486) / Jannik Marius Scheuble	Semester: 3
Date: 14.12.2020	Group:10
Instructor: Lukas Ewecker, Markus Wachter	

1. Executive Summary:

While we are living in difficult times at the moment, Covid-19 has taken control of our lives, and we must limit our social contacts as a result of the pandemic. We, as a young start-up, received a order from a customer. The order included that we have to build a robot who is able to win in a tic tac toe game.

This is the written elaboration about the process and our thoughts of the Tik Tak Toe challenge, in which we gain practical experience in programming and can apply our programming knowledge learned in the lectures. The Project is based on an application on SCRUM for software designing.

SCRUM is a popular approach to project management today, but its origins are in agile software development. This tool helps us as a guideline for the realisation of a software-based project like our self-playing robot. Therefore, we have been tracked our process weekly, which also helped us a lot to keep a clear structure. Part of the weekly Scrum approach was that every sprint we should have a short presentation about the last sprint, where we should present our progress. This also served as a guideline to have an overview of how far the other groups are, so that you could assess your own progress well. And mainly as an overview for the client, allowing them to see the current status of the project.

2. Table of contents

3. Introduction:

3.1 Introduction of the team:

Our team consists of three members which are Felix Schnoor, Julian Maier and Jannik Scheuble. In this project Jannik was the project owner. We have never worked in this group constellation before, and also are very new to programming. Nobody of us had experience working with the scrum project management method, also no one of us got experience on how to organize a project. Regarding all “disadvantages” we were motivated to tackle the Tik Tak Toe project.

3.2 Introduction to the project task

The task was to write a program that allows the robot to play Tik Tak Toe against another robot or in this case just any opponent. The real challenge is that everything has to work without any user input via the PC. This means that the sensors need to be used to inform the robot when it is his turn and where the opponent made its move. This also means that the robot needs to know who starts the game. The end goal is for the robot to successfully complete the challenge and run consistently without major issues. The task should be completed within the given five weeks with the help of the Scrum method which consists of 3 one weeklong sprints, 1 week for the project setup and 1 week for to complete everything before the Challenge. The final challenge is then livestreamed to everyone. Here we have 10 minutes to play 2 games against our opponent. Throughout the project we struggled with the impact of Corona and its lockdown, so we met online via Microsoft Teams and held our meetings and virtually more often than we would have liked. But in the end, we managed the balancing act between virtual meetings and real meetings quite well.

3.3 Technical and other preconditions:

The Technical preconditions for the project was to build the Robot and modify it in a way that it can play efficiently and pick up the block. We also need to decide which sensors we want to use for our robot, because there are more sensors than sensor ports. Other preconditions then technical are the rules that are set for the project. The most important of these is that the robot has to work without any PC input and cannot be moved after the game starts. The robot also needs to be detected the end of the game by itself and how it ended. It also needs to be able to end the game before all nine terms are done if one player wins. The only way to give the robot inputs is with the help of the sensors which can be used. But here it is important, that the robot has to stand still while it receives its input. Once the robot is going, we cannot interact with any of these sensors until it is another turn. There is also a condition on how the field has to be designed in a specific way. All squares have to be the same size and the distance of the starting line to has to be 1,5 times the length of one square. In addition to that the robot also has to start somewhere behind the dashed line but otherwise it does not matter where the robot starts. There are also rules for the block placing, which mandated that the robot has to place a block each turn. The blocks can be moved by the robot during terms but need to be back in their place when the other turn starts. If the blocks are not placed precisely it counts for the square where it covers the largest area as the square. The game is played with the normal Tik Tak Toe rules were the first player that gets three squares in a line wins. The game is played with alternating turns. All this has to be done with the given Project Hardware which is listed in the next point.

3.4 Project Hardware:

Ev3 Brick:

The Ev3 Brick is the processor of our little robot. Through this brick he is able to execute and calculate moves. The block got 4 inputs (1-4) and 4 outputs (A-D). The brick is also able to connect to computer via cable, W-Lan and over Bluetooth. There also is a display where you can navigate through the interface of the robot.

Colour Sensor:

The colour sensor is able to detect colours like the name already tells it. He can differentiate between 7 different colours. For every colour there is a colour value, for example green = CV 3

Touch Sensor:

The touch sensor can differentiate between being pressed or not. If the Sensor is pressed it outputs a value of 1 if it is not pressed it outputs a value of zero.

Ultrasonic Sensor:

This sensor is able to detect objects in the room. It generates a sound signal and measures the time which the sound takes to the object and back. It takes the time that it took to the object and back and calculates the distance to the object.

Gyro Sensor:

This Sensor can be used in 2 main modes. These are measuring the angular velocity or the current angle. It is mostly used to measure the rotation a round one axis. It has an accuracy of +/- 3° on a 90° turn. A clockwise rotation is positive while an anticlockwise rotation has a negative value.

Motors

For the project we got two different types of motors. There is a medium motor and a large motor. We were given two large motors and one medium size motor. The large motors are used to drive and the small motor to pick up the blocks. The commands for the smaller and bigger motors are the same.

3.5 Project Software:

In the lecture we learned the basics of programming with Python 3, accordingly we had to use the programming language Python 3 for the robot. This allows us a extensive and task-specific programming of the robot. In order for the robot to read the Python files, it has to rely the software platform ev3dev. This is a tool basing on linux, for controlling steering motors and sensors, that’s the reason why we have used the ev3dev for the challenge. This also means that you have to setup a custom interpreter which works with this. We also have to import the ev3 program on the computer to test this feature. Because the system is Linux based and we work on Windows the Project files always have to start with this in the first two lines.

```
#!/usr/bin/env python3
import ev3dev.ev3 as ev3
```

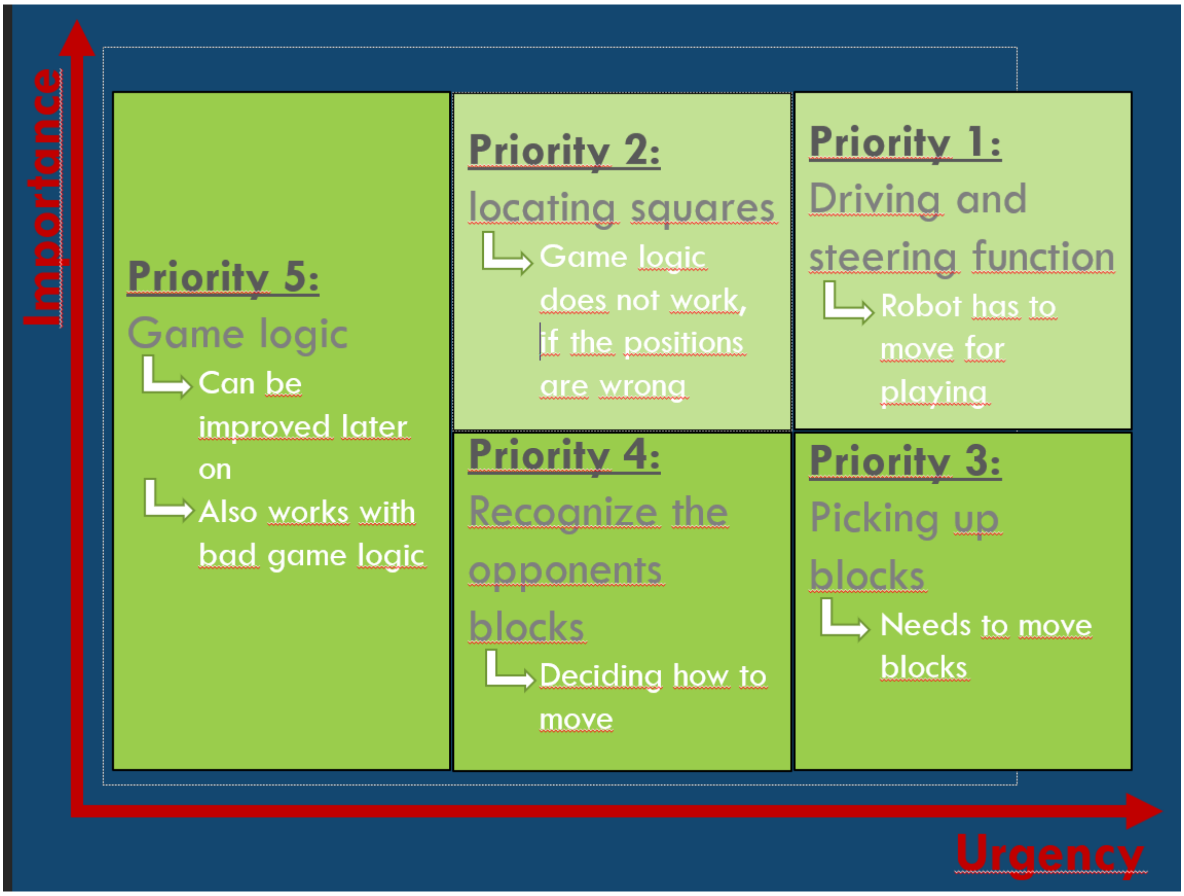
4. Plan to approach the project:

4.1 Analysis: Plan to tackle the problem:

Our plan for talking this project was to start with the driving and steering function which would help our robot to navigate to the correct field, and move the block there. After we accomplished that we plan to start programming a to create a playing field. We also have to program a function that remember the moves that were made. In addition to that we need to know when the game ends and how it ended. After all this is done, we start with the minimax Algorithm for the robot. Minimax are often used for games like Tik Tak Toe, Backgammon, Go, Chess and many other games. The goal of the minimax algorithm is to is to maximise the players possibility to win and to minimize the opponent’s possibility to win. This leads the robot to play with an ideal or nearly ideal strategy. With our robot we plan to implement this algorithm in a way were in the beginning the other players input will still be entered through the PC. Our thought process behind this was that we made life easier for ourselves by not having to focus on several things at the same time, but that we could put our main focus on the strategy. It was also more logical and easier for us to understand what our code actually looked like at that point in time. After this works fine, we want to adapt the program in such a way, that the input of the opponents turn can be done with the help of the sensors. This will probably be done via the colour or the touch sensors because they are the easiest to interact with on the robot without having to touch or move the robot in any way which is part of the requirements and preconditions. Another challenge is that the robot’s motors are not reliable to always work the same way. That is why we want to avoid hardcoding movements and instead use sensors to navigate the robot more reliably. Because of this unreliability we probably also have do some optimizations to correct any faults or make adjustments to the way our program works if needed. We predict that we will need to some off these adjustments which we plan to do in the week before the challenge. We hope to be finished a bit ahead of schedule so we can focus the last week on finishing and handing in the final document before the start of the challenge. For the challenge we want our robot to win the game or at least always draw because of the nature of Tik Tak Toe as a game. It would be nice to win a game but that mostly depends on the opponent's robot and if it plays in a smart or a random way, because normally if both players do not make a mistake, Tik Tak Toe will end with a draw.

4.2 Prioritization:

Our prioritization for the project was depending on how they affected the success off the project and depending on their difficulty and the effort they take to program in the project. We also may have to adjust this depending on the progress with our project. This following diagram shows our priority's weighted by urgency and importance. This also shows the reasons for their placements in the chart.



4.3 Using Scrum:

We are using scrum methods for the whole of the project to plan and manage the sprints. Every meeting we discuss which points we have to do during a sprint to remain on schedule.

5. Project documentation

5.1 Project Setup:

5.1.1 Sprint Goals and plan:

The first week was for the project setup, were we picked up the robot for the project from Mr. Wachter. Before starting to build the robot, we had to control the inventory and make sure all parts are there as expected. We then started to assemble the robot and connect the sensors and motors. After that we had to set up the work environment to work effectively with the robot. We decided to work with “PyCharm Professional” to set up our work environment. We also had to set up the project interpreter so we could work with the commands for the ev3 in PyCharm otherwise we could not work with the robot effectively. All in all, the project setup was done successfully and without major problems occurring. But before we could start with the real project, we also had to pick up the blocks. We also initialised our git repository so we can safe and exchange our code with each other and to corporate more effectively with each other. We also prepared our first presentation for the project setup where we picked our project vision and started to plan our strategy and backlog for the first time. We also did the effort planning of our project.

User story: The robot needs to be designed, to be able to play Tik Tak Toe.

5.1.2 Tests:

We tested our Setup by transferring the marsrover.py file via WinSCP onto the robot and made the file executable so that we could test if our setup and the basic functions worked successfully.

5.1.3 Bugfixes and Changes:

There also were some problems for example we misunderstood how the effort estimation should be done. We thought that the effort estimation should show in which timeframe we want to program the different features. What was really meant was how many hours we would need to program a certain feature. This would allow us to track better if we are on time or whether we are behind the schedule.

5.1.4 Retrospective:

This first week everything worked pretty good without major problems and relatively good cooperation between the Team members for working for working together for the first time in this group constellation. In this first week for the project setup, we did not set a smart goals jet. We started to think about it for the first sprint and how we want to work in the group together.

5.2 Sprint 1:

5.2.1 Sprint Goals and plan

The first goal of this sprint was to correct some of the mistakes of the project setup. Because our first effort estimation was wrong, we came up with a better one this sprint. We estimated that the project would take 90 hours in total with 10 hours that we already spent for the project setup and 20 hours for each of the following 4 weeks. To measure the effort and velocity of our project we decided to use a burndown chart where we visualized our weekly progress to see whether we remained on time this week or lagged behind schedule. The next goal for the first sprint was to code the driving, turning and steering functions so we could access them for navigating the board. But before we could do that, we decided on a bord where each square is 10 cm. So, the total board was 30x45 cm. We then programmed the picking up blocks function where we used the colour sensor to locate the blocks. We had to figure out how to use the small motor to hold the block and how to let it go. We decided to do the dropping of Blocks function in a new function. We then coded a function that allowed us to navigate to certain squares with the help off our previously programmed steering and driving functions. We drop the blocks by just stopping the small motor. We also planned to program a function, that remembers the moves that that were made by the robot and later also the opponents move. After testing and fixing most problems we also pushed our code onto GitHub to protect it from getting lost. After that we pushed our code to GitHub, we noticed that our project files did not appear in git status again, so we could push them again when we made changes to the initial code. Another Problem that we encountered, was that only the person that has the robot can test if the code works with it. This resulted in some problems with testing code but all in all everything worked just fine.

User story: The robot has to be able to drive on the field.

5.2.2 Tests:

While testing the dropping of blocks function, we encountered the problem that the impact of the Blocks falling would sometimes cause them to jump into another square. Also, while testing the picking up blocks function, we discovered that the colour sensor is very unreliable when it is on a different hight then the middle of our block. There also was a Problem with the grabber picking up the blocks. The blocks would sometimes fall out of the grabber based on the way they got picked up. This is a result of the block structure and the smooth surface of the Lego. We also implemented the robot control function. The test for this function was that we tried to drive to a specific field.

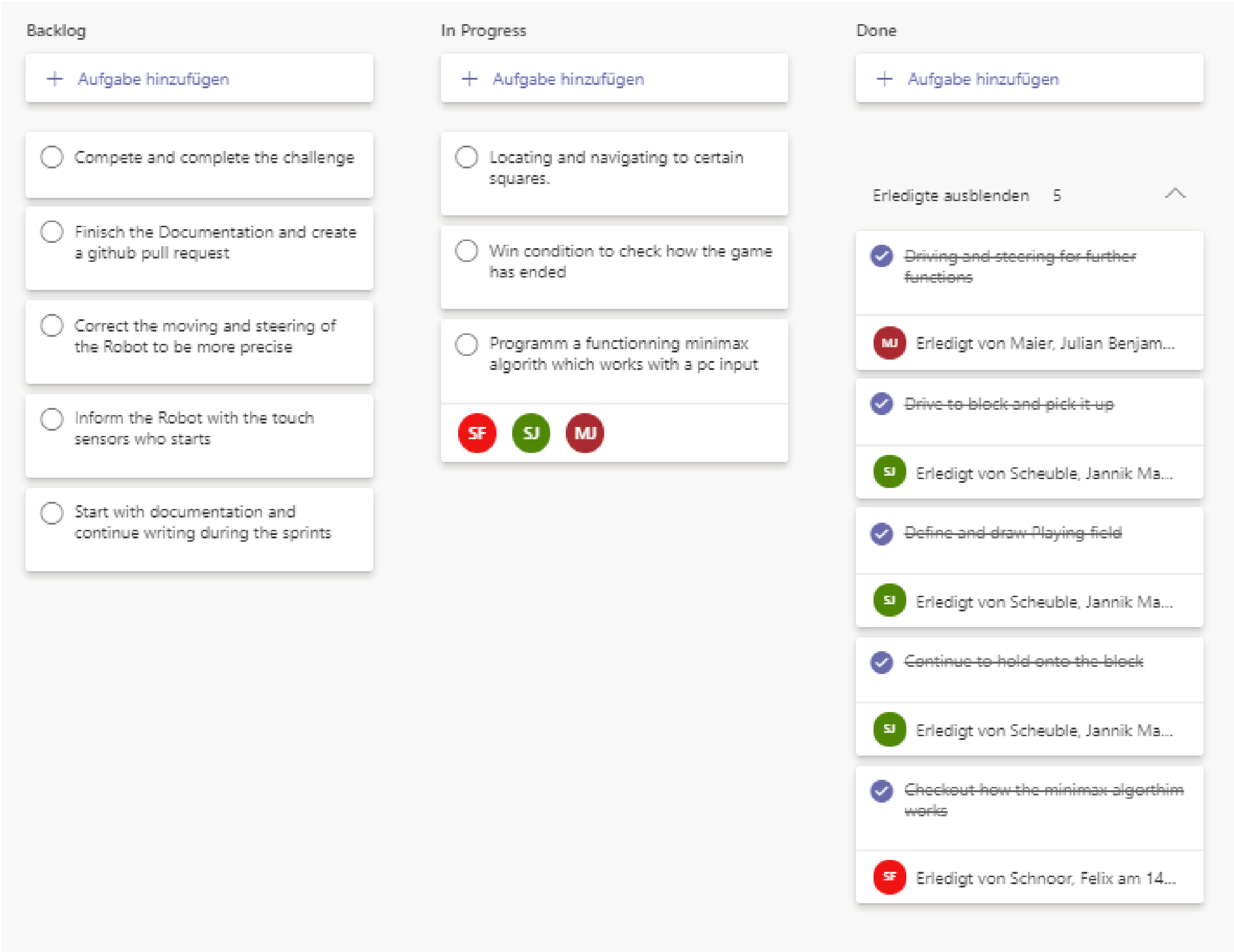
5.2.3 Bugfixes and Changes:

That is why we decided to enlarge the squares to 15 cm to prevent this. This lead to our playing field now being 45x67,5cm. That also lead to us having to modify our driving functions so that the distance is correct. For the colour sensor we adjusted the hight of the sensor that it is on the same hight as the coloured block. We also discussed the possibility off lowering the grabbing arm to remove the need for a block dispenser an instead we could just place the blocks in one row. This would probably also allow us to use a smaller field, because the blocks wouldn't be able to gain the momentum that allowed them to jump into another field. We also found a solution for the Blocks falling out of the claw. We fixed this by simply continuing to move a bit further, so that the block will always be at the end of the blog claw. We didn't find a solution to the git status in this sprint yet, so we decided to just wait with pushing to GitHub until we found the solution to this problem.

5.2.4 Retrospective:

In the first sprint we implemented the Kanban board on "Microsoft Teams" to organize which tasks are currently done, not stared yet or finished already. Like this we could assign the tasks to a certain group member and give him the responsibility of the task. The problem for this sprint was that we set our goal way to far from the realisable point. This was a reason on why we already fell back on our planned schedule in the first working week. We also underestimated the effort of the tasks we had to do. As a result of this we decided that we have to lower our tasks per week so we can be more realistic and focused on the goal. The highlight for this sprint is that we accomplished the first two functions, which are picking up the block and driving.

Kanban board:



5.3 Sprint 2:

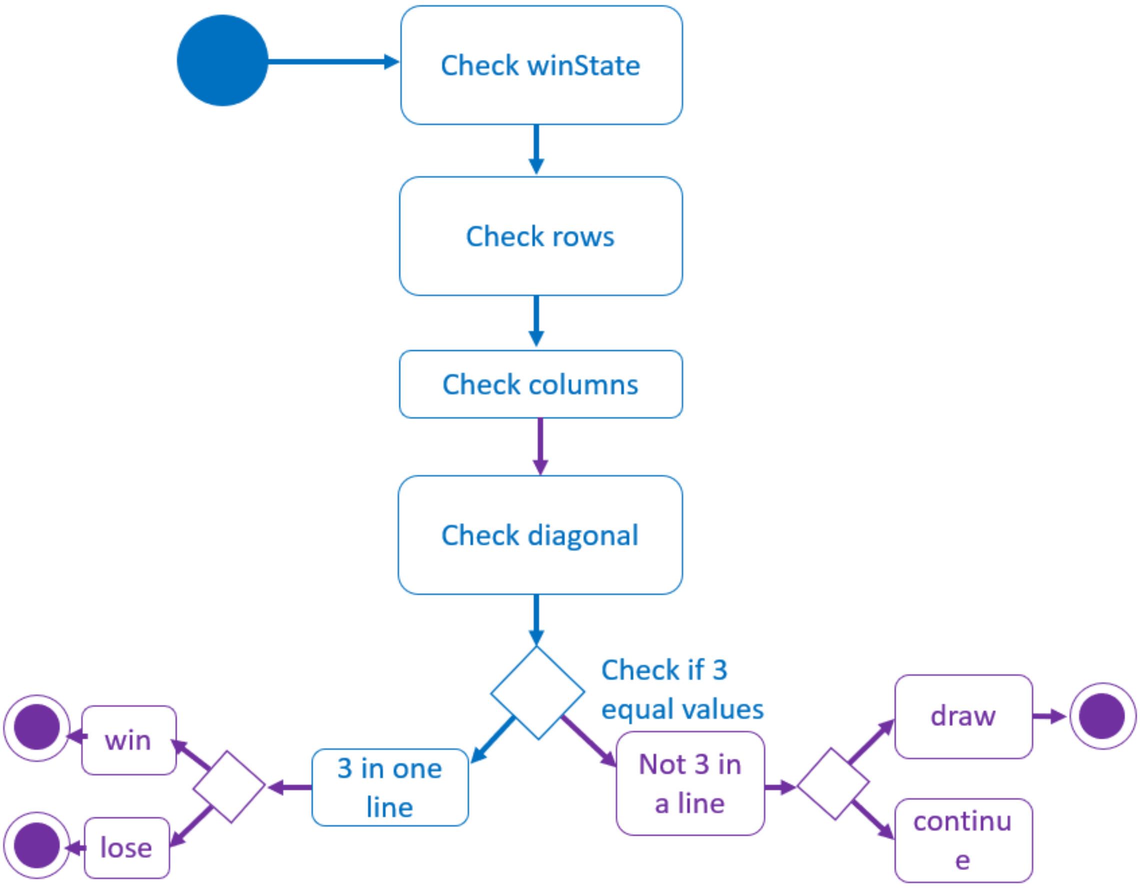
5.3.1 Sprint Goals and plan:

At the beginning of sprint two we looked back at our sprint one and analysed our mistakes. We made sure that these mistakes will not appear at this sprint. After that we found the solution to our git status problem from sprint 1. We noticed that this problem fixes itself. If you change something in the file, it reappears as a modified file in git status and can be added to git commit again. For this second sprint we planned to do the win condition and finish the functionality to let the robot remember where it dropped its blocks of from the last sprint. To remember where the robot dropped its blocks, we were able to use 3 arrays of length 3 in another array to represent our board. Each internal array represents one row of the board. To simplify interaction with the board and to prepare for future functionality changes of the board, we decided to put the board in its own class. We decided to use the excess time if available to look into the minimax algorithm to prepare for the second sprint. Because we lost some time during the first sprint, we wanted to get to our planned goal to catch up on our schedule. We actually had a bit excess time and tried to inform us about the minimax algorithm and how we can implement it in our code. The minimax algorithm is our tool to play the game with certain “game knowledge”.

User Story: The robot has to know when and how the game has ended.

5.3.2 Tests:

We tested our win condition, by confronting it with different scenarios, i.e. different board states. The win condition checks if three squares in a row, column or diagonal have the same value (unequal to 0, as 0 indicates an empty square). If this is true it announces who won by returning value 0 (for player 0) or 1 (for player 1). If nobody wins, it returns –1 and the game continues with the next move. If all moves were played and nobody won, the game it is a draw and the function returns 2. The win condition needs to be checked every time after the fifth move. For the sake of simplicity and making the code easier to look at, we decided to check the win condition every move, since this function is quick enough to be negligible.



5.3.3 Bugfixes and Changes:

While testing, the robot had a few issues when navigating the physical field. This wasn't a big problem because we could fix this by adjusting the code a bit.

5.3.4 Retrospective:

The goal that we gave us for this Sprint, was that we have more frequent meetings via "Microsoft Teams". This was necessary because we didn't achieve our goals in the first sprint. We planned to measure this by meeting us at least three times a week to get us an overview over the project status. Like this we wanted to make sure we achieve our goals as planned. We also prioritised the robot project over other subjects while doing the second sprint to make sure we get caught on our schedule. After setting our retrospective goal we were able to get to our planned time zone and even a bit further that. This was also the highlight of the week. The lowlight for this sprint was that one of our team members had to go into quarantine and we couldn’t meet up with him in that time span.

Minimax Algorithm

The minimax algorithm is an algorithm to evaluate the best possible move for the game. It looks at every move and evaluates which is the most beneficial for the robot. He generates every move and gives the moves certain values starting with the leaf nodes. If he gets a disadvantage due to a move, he gives it the value (-1) and if he wins with a move, he gives it the value (1) and the value (0) if he generates a draw with its move. The values get generated first at the leaf nodes (leaf nodes are Nodes where the game ends). From there on the value gets generated upwards. If it’s the robots turn, he takes the highest calculated value, because this gives the most advantage for him. Like this he can weight up his move and also rate it and decide on that.

5.4 Sprint 3:

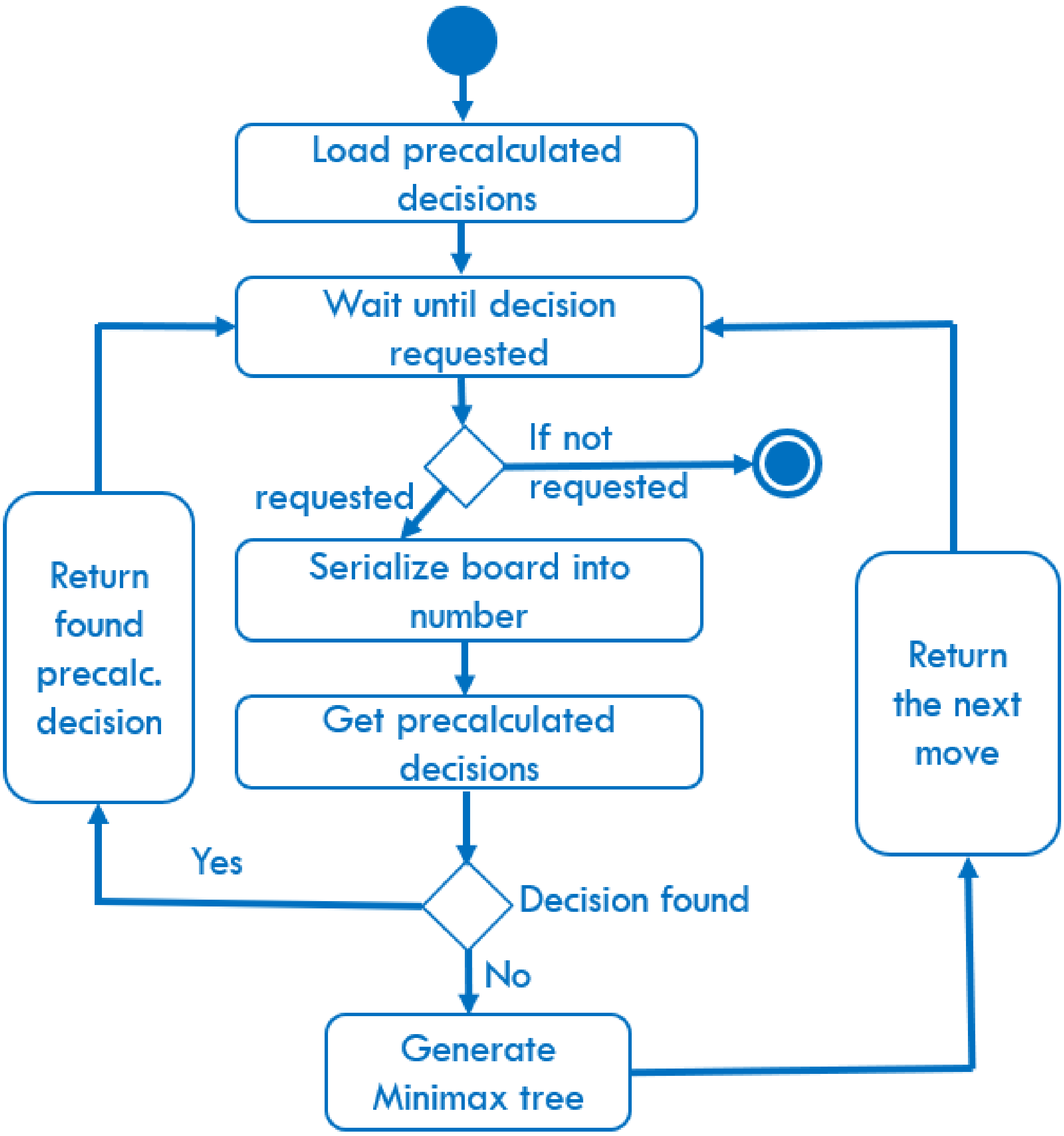
5.4.1 Sprint Goals and plan:

After we had thought about the game logic in sprint 2, and had already informed ourselves about the minimax algorithm on the internet, we set ourselves the goal for sprint 3 to program the minimax algorithm. Another goal was to outsource our win condition function into our board class, as this makes it easier to interact with our board. Our third goal for the sprint was to think about controlling the robot without the help of a computer. In the rules of the challenges, it is already given that we have to make inputs through the sensors. We thought that it would be best if we make the inputs over the touch-/ or colour-sensor.

User story: The robot should make the best move depending on the situation, with the help of the minimax.

5.4.2 Tests:

Testing the minimax algorithm was certainly one of the most fun tasks during the whole project. If we wanted to test something on the robot there always was time lost during the data transfer for every change. This and also the inferior performance of the robot compared to a PC led us to instead implement a game loop for the python console so that we were able to quickly iterate code changes and fixes for the ai. Through that we were able to test the ai. We controlled our input via the computer. Consequently, we wrote a to-do list together with the tasks that we still have to do.



5.4.3 Bugfixes and Changes?

We were able to fix this error somehow, but we still do not know why the error occurred or what exactly caused it. We then quickly pushed our code to GitHub to secure it from getting lost. Another problem that occurred was that, while we made the first moves with the minimax algorithm our robot crashed due to a lack of processing power. This was happening because the calculations of the first moves were way too big for that little brick. Therefore, we generated a file called “ai_first_moves.ai” containing precalculated decisions. With this file it is possible to play the first moves. Since all board states are uniquely identifiable by a number, we were able to use a number as a key to a dictionary containing precalculated decisions for certain board states. To generate this number, we simply need to represent a given board as a base 3 number. This is done by concatenating the individual rows of the board giving us a number in base 3 (since all squares of the board contain numbers from 0 to 2). This number can then simply be transformed into a normal decimal number for use in our dictionary.

5.4.4 Retrospective:

We were able to achieve our retrospective goal from the last sprint. The goal was that we have more frequent meetings via Teams. In the third sprint we didn't make major changes in our retrospective goals. We did it like the sprints before. We assigned the tasks that are needed to be done to the group members and tried to fulfil them. It occurred a problem considering our code and we thought that we lost our code. This was the lowlight for this sprint. The highlight of this sprint was that we could play against our little robot. He has beaten us every time or got us to a draw.

5.5 Week until challenge:

5.5.1 Goals and Plan:

This was the final week before the challenge, were we had to finish our documentation and complete the challenge by the 15 of December 2020. We still had problems with the motors not moving in the correct way. We also experienced some troubles adapting the code to working without the computer input. This was especially challenging for working with the touch sensors which we had never used before. We also had to find a way to notify the robot of the opponents move. This could be done via the touch sensors or the colour sensor. In both cases we first have to input the row for the blocks and then the columns. Those inputs have to be done in two steps because of the way our robot works. We first get the row number and then the column number of the opponents move. We then make sure, that we entered the right data. We also have to fix the moving of the robot. We lowered the claw of the robot to end the need for a complex block picking up station. We also had the hope that this would allow us to go back to a smaller field where we have to drive less which also reduces the error the robot makes, when moving because it has to move less and can move at a slower pace to reduce the effect of the motor deviation. We went back to the original field size of 30cm by 30 cm.

User story : The robot should be able to play a whole game against an opponent.

5.5.2 Tests:

The final Tests of the project is testing the robot and playing complete Tic Tac Toe Games against a person and making the file executable for use in the challenge. This will still require a lot of effort to fix this code. To prepare for the challenge. We also have to test the Input via Sensors thoroughly to make sure, we do not accidentally enter the wrong values.

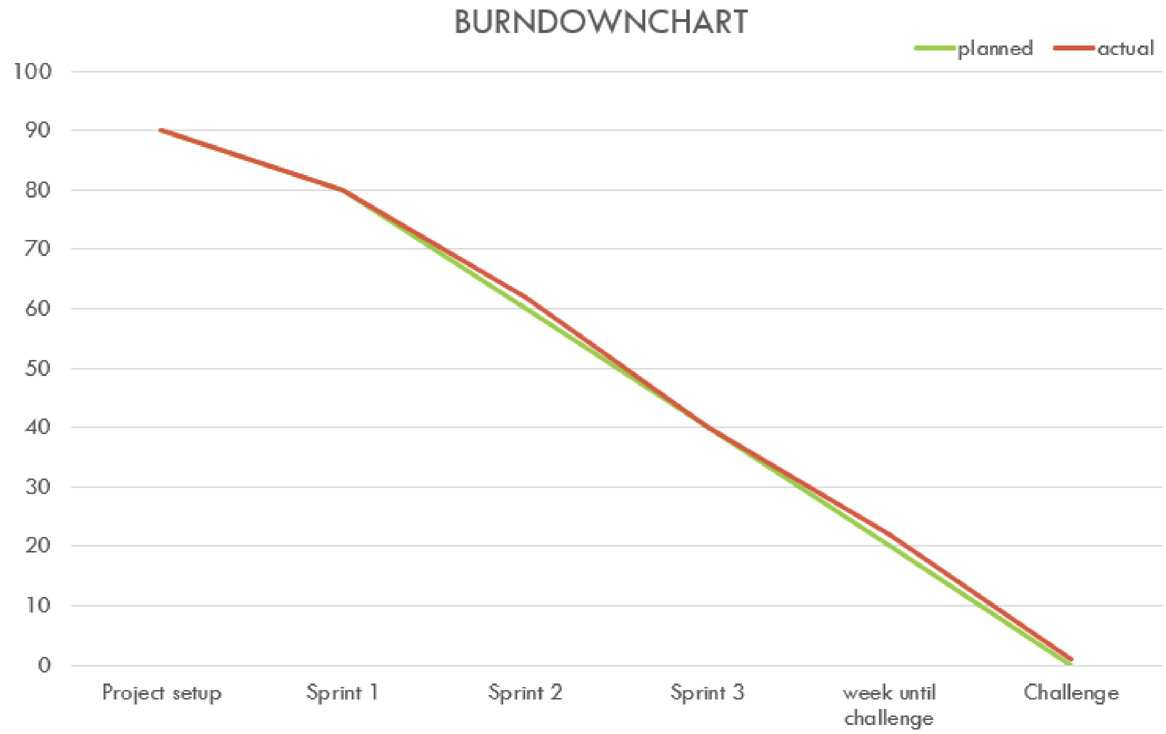
5.5.3 Bugfixes and changes:

In the last week we were finally able to correct the inaccuracy with the steering. This was possible because we slowed the motors speed. With that there were smaller differences between the speed of the motors. Because of our smaller playing field, we also had to adjust our driving functions. To prevent that wrong values are entered we implemented a function that allows us to error check the input and also informs us, if the entry is invalid.

5.5.4 Retrospective:

For the last week of the challenge, we tried to set new retro goals to finish our documentation on time. We started “splitting” the documentation between us, by that we mean that we work together with equal effort. The goal was to write at least 5 sites per person to move forward with the documentation. We all assumed that this should be possible for one week. This task was pretty important for us because the documentation determinates 40% of the mark. We also met a few times in this week, to make sure that our robot will run correct during the challenge. The lowlight of this week was that our robot still had issues with driving inaccurate during our tests.

6. Summary of the Project weeks:



We planned with 90 hours for the whole project. In the first week for the Project Setup we remained on time as planned and completed all tasks. In the first sprint we lacked a bit behind our planned schedule, because were not able to create the function for remembering the moves that made. In Sprint 2 we were able to catch up to the schedule by working a bit harder and integrating the remembering squares function with the board creation function and the win condition. In Sprint 3 we accomplished all our goals but needed a bit more time than we actually planned. In the Final week until the challenge we worked quite hard to finish the robot intime. But the disappointing result of the challenge was that our robot did not complete the challenge correctly. For this reason, we did not run the burndown chart line to zero, which would mean that we successfully completed the challenge. We were not satisfied with the result and we want to improve our robot until the 2nd game, so that the graph arrives at zero.

7. Project Reflection:

Sprint 1:

We did not accomplish all the goals we set our self for the first sprint, this was mainly because we underestimated the effort that it takes to program the different features and coming up with an idea how to program them. That ultimately was the reason why we didn't find a way yet to remember the moves, that the robot has made. The main difficulty with this was that had not done anything very similar to this task yet. The difficulty was to remember which moves could still be made and how to differentiate who made the move and which move was made. We also severely estimated the effort it would take to find a solution for this problem. This caused us to be behind our own schedule. Because of this we decided to set goals that were smaller and more achievable which also will help us to measure our progress more accurately and more consistent. This was our lowlight for the second sprint in addition to our problems to coordinate working online. The highlights of our second sprint was that the robot was able to pick up the blocks and drive to the squares were it then drops off the blocks. We also got the feedback that our burndown chart for the effort and velocity estimation should show the whole project and not only the Sprint to better visualize whether we are on time or not. All in all, Sprint 1 was a pretty good sprint but because of us being pretty new to this scrum method and actually programming with the ev3 for the first time we encountered these little problems. We expect that these problems will not happen again in Sprint 2.

Sprint 2:

If we draw a conclusion about Sprint 2, we can definitely say that it was a very successful sprint in which we achieved a lot. We were able to complete all the tasks as planned, which were programming remembering the squares function, win-condition, board creation and informing us about the minimax algorithm. Furthermore, we were able to catch up again in the sprint due to high engagement, so we were back on schedule. The lowlight of the sprint was definitely the moment when we let the robot driving on the game board for the first time and we noticed that he was driving slightly inaccurately. Therefore, we have put this issue on the to-do-list.

Sprint 3:

Looking back on Sprint 3, we can say that it was an interesting week in which we achieved a lot. The completion of the game logic code counts as a big milestone. And after running a few successful tests, this was certainly our big highlight of the week. However, not everything went smoothly during the week. Probably our biggest lowlight of the whole project was when we almost despaired when we had problems with saving and pushing the code to GitHub. This error shortly led us to believe, that we had lost or code because all project files that seemingly disappeared from the project folder and lead to us seemingly losing our code. Fortunately, that was not the case and we were able to solve the problem, giving us further motivation for the task.

Week until challenge:

This week was stressful again as we underestimated how much effort it would take to implement the missing input features and make the robot move in a correct way. Implementing the feature to input the opponents move went well. But correcting the steering function was pretty hard and we noticed that it probably would not be possible to make the robot run correct. Because the faults add up in the end. We could have made this better if we had used the gyro sensor. But this far into the project it was too late to make major changes like implementing and successfully using a new sensor. We also had to finish our documentation until 15 of December 2020. This lead to us having to split our focus on the documentation and fixing our code. We decided to focus more attention on the documentation, because it was more important pointwise then the actual challenge.

7.1 Summary of retrospectives

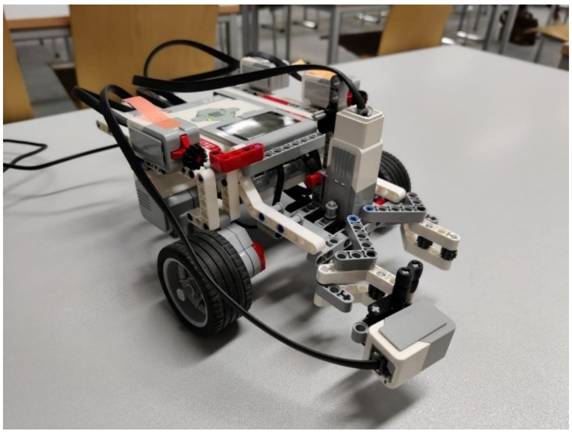
Our team started setting retrospectives goals in the first sprint week to improve the quality of the group performance. The first thing we implemented was the Kanban board. We used it to assign tasks to group members to make them responsible for the tasks. In the 2nd sprint we needed to get on our schedule. We solved this problem by planning three meetings per week via “Microsoft Teams”. With the meetings we were able to keep track of our project progress and also made a little head start. At the 3rd sprint we didn’t make any relevant changes and continued like the sprints before. In the last week we mainly focused on the documentation and some miner fixes on the code and the robot. We “splitted” the documentation and worked on it with equal effort. We used a Google Drive Word document to be able to write simultaneously on one file.

7.2 Results of the retrospective

With the weekly retrospective goals, we were nearly always able to improve our internal working performance. Through that method we were able to look over our mistakes from last sprint and make sure they do not occur again. We also improved our performance with implementing certain retro goals such as making several meetings per week. With them we were able to meet us up to three times per week.

7.3 Result of the challenge:

We were not able to succeed in the challenge. The problem with the inaccurate turning was still occurring. At the start of the challenge our robot turned a bit too far and missed our block dispenser. This was the reason that the robot was not able to pick up the block. Because of this error we lost during our first move. We plan to fix this problem till the next date of the challenge when we can attend the second game. Even though we did not get points for playing in the challenge, yet we got three bonus points for creating a playing field that was created according to the requirements.



7.4 Review of Scrum as a project management tool:

Although we did not have that much prior experience with the scrum project management method, we were able to get used to it quite quickly. The sprints where really good for us to analyse where we are considering the time. We already got an advantage of it in the first sprint. While we fell back on our schedule in this week, we already became conscious of it because we had to review our progress. This helped us understand why this method is so commonly used. For us it really helped us to measure our progress and understand how far we are into the problem. The retrospectives helped us with organizing our group and also improving how we work together effectively. Tools like the Kanban board in Microsoft Teams were also very helpful to have a much better overview about task to do.

8. Annex:

8.1 Declaration of Originality:

Declaration of Originality

Subject: Project in Computer Science for Engineers (WS 2020/2021)

We

Name 1: Felix Schnoor

Name 2: Jannik Scheuble

Name 3: Julian Maier

affirm that

- we wrote this lab report by ourselves as a team;
- we all were involved to an approximately similar extend in the lab work;
- we as authors all know and agree about the content of this submission;
- we did not use other than the denoted sources and aids;
- we have identified all textual and analogous sites for the report.

Signing this declaration means that you are not allowed to plagiarize, nor identify a student as co-author, if he/she did not participate actively in carrying out the lab experiment and writing the report.

Doing so, is a serious scientific misconduct and will lead to the fact that the whole team will not pass the lab.

Signature Name 1	Signature Name 2	Signature Name 3
F. Schnoor	J. Scheuble	

8.2 Image directory: