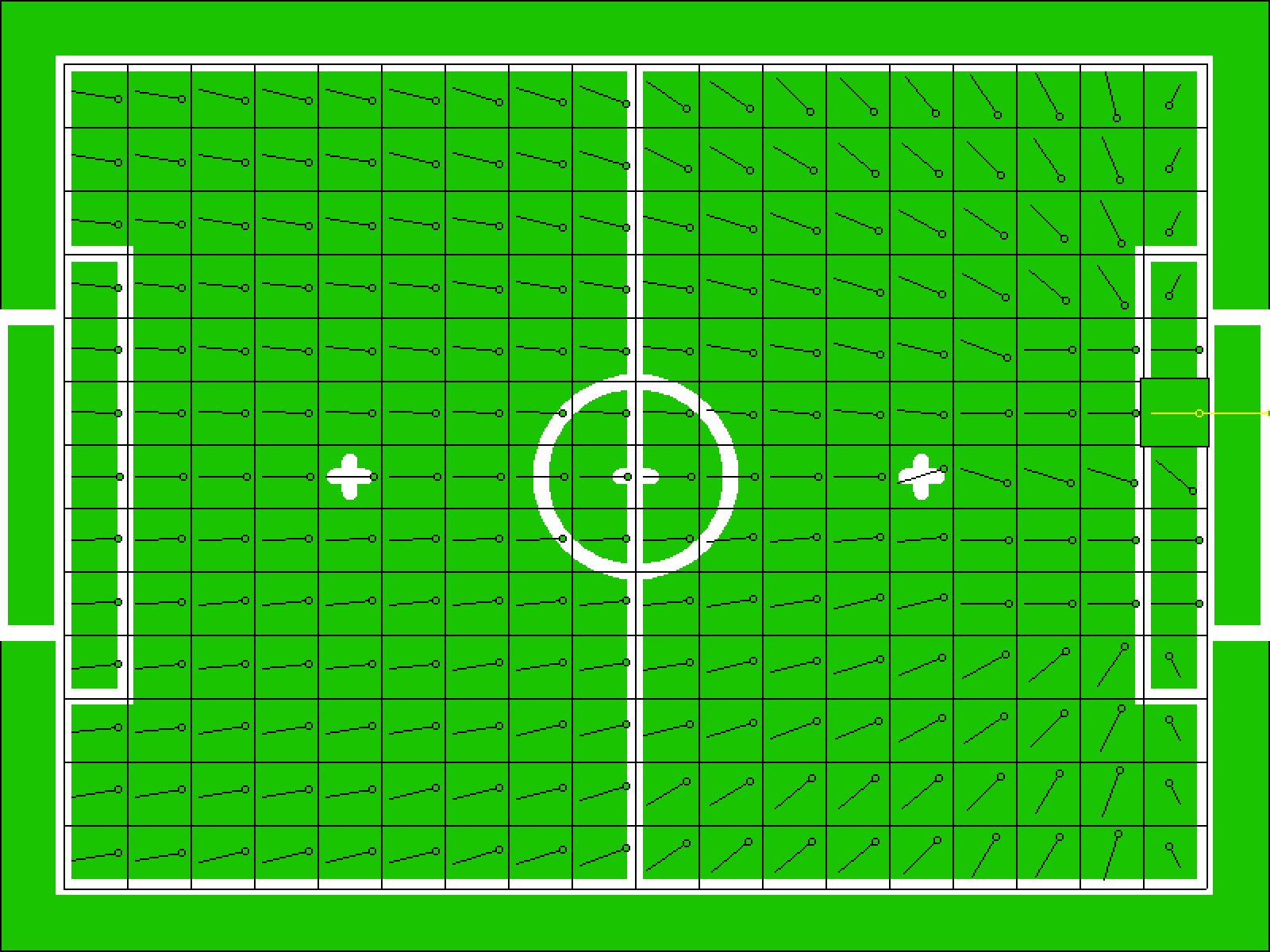
ELSIROS

Entry\_Level\_SImulation\_of\_RObot\_Soccer

Robot programming hints

1. It is not advised to play games between completely equal teams. In most cases you can observe non-interesting game development with rare goals and useless struggling. Therefore, we include 2 styles of playing algorithms: normal and old style which was practiced in year 2020.
2. Real Robots use OpenMV H7 smart camera as vision sensor and onboard computing module. This is single core controller with Micropython bare metal programming. This means that controller is not capable to provide walking engine and camera vision simultaneously. In order to update information about ball position and self-localization robot has to stop into stabile stand-up position and move head to various positions to observe surroundings. During head moving ball position can be red in case if ball is in visible sector of camera. Camera have 46 degrees of aperture.
3. During observation of surroundings at stand-up position of robot camera catches additional information about robots’ localization from objects like goal posts, field marking and green field border. The more pictures are taken the better accuracy of localization.
4. Robot can detect obstacles from vision sensor. Obstacle avoidance algorithm is included into path planning, but it is not perfect in all aspects. There is not implementation of kicking ball strategy with accounting possible obstacles at ball path. Detected and updated data about obstacles are stored in list self.glob.obstacles
5. Communication between team members is legal by rules through UDP messages. Communication is not implemented in current game strategy, but it is allowed to be developed by teams. Communication inside team can help to organize team play. ELSIROS API provides functionality for messaging between team members.
6. Coordinate system of field for purposes of strategy is different from absolute coordinate system of field. For purpose of strategy own goals are located at part of field with negative X coordinate, opponents’ goals are located at positive X coordinate. Positive Y coordinate is at left flank of attack, negative Y coordinate is at right flank of attack. Yaw heading is zero if it is directed from center of own goals to center of opponents’ goals. Yaw is changed from 0 to math.pi with turning to left from zero direction. Yaw is changed from 0 to -math.pi with turning from zero direction to right. Z coordinate is directed to up with zero on floor.
7. Normal “forward” player uses predefined strategy formulated by vector matrix. Matrix is coded in file strategy\_data.json This file is readable and editable as well as normal text file. There is a dictionary with one key “strategy\_data”. Value of key “strategy\_data” is a list with default number of elements 234. Each element of list represents rectangular sector of soccer field with size 20cmX20cm. For each sector there assigned a vector representing yaw direction of shooting when ball is positioned in this sector. Power of shot is coded by attenuation value: 1 – standard power, 2 – power reduced 2 times, 3- power reduced 3 times. Each element of list is coded as follows: [column, row, power, yaw]. Soccer field is split to sectors in 13 rows and 18 columns. Column 0 is near own goals, column 17 is near opposed goals. Row 0 is in positive Y coordinate, row 12 is in negative Y coordinate.



1. During game player can take 4 roles: ‘forward’, ‘goalkeeper’, ‘penalty\_Shooter’, ‘penlaty\_Goalkeeper’. For each role strategy code is different. Launcher module chooses role of player to launch depending on number of player and secondary game state. In case if number of player is 1 then appointed role will be ‘goalkeeper’ or ‘penalty\_Goalkeeper’. In case if number of player is other than 1 then appointed role will be ‘forward’ or ‘penalty\_Shooter’. In case if secondary game state is ‘STATE\_PENALTYSHOOT’ then player with number 1 will be appointed as ‘penlaty\_Goalkeeper’ and player with other number will be appointed to role ‘penalty\_Shooter’. Default public player controller strategy appoints player role ‘goalkeeper’ to player with number 1 and ‘forward’ to player with number other than 1 in all other secondary game states. Teams can modify strategy and use various roles depending on secondary game state. According to current game controller there could be following secondary game states: STATE\_NORMAL=0, STATE\_PENALTYSHOOT=1, STATE\_OVERTIME=2, STATE\_TIMEOUT=3, STATE\_DIRECT\_FREEKICK=4, STATE\_INDIRECT\_FREEKICK=5, STATE\_PENALTYKICK=6, STATE\_CORNERKICK=7, STATE\_GOALKICK=8, STATE\_THROWIN=9, DROPBALL=128, UNKNOWN=255