Autonomous Systems Practical Assignment 5: Goal Alignment and Kicking

Sho Cremers (S3135144)
Sukhleen Kaur (S3157423)
Group 19B

April 1, 2018

Introduction

The goal of this assignment was to get the Nao to search for the ball, approach it and then align with the goal and kick the ball into it.

Methods

Programming the Nao for this goal was done in Python language. The use of the package `basebehavior.behaviorimplemtation` was made as well with the use of the package `rospy`.

The branch name: `Align_Cremers_Kaur`

The run instructions are as follows:
4 terminals are required (in order of running):
- Terminal 1: `roscore` (in order to be able to run the behavior)
- Terminal 2: `roslaunch borg_nao colorblob.launch` (in order to see what the Nao sees)
- Terminal 3: `roslaunch borg_nao borg_nao.launch` (in order to see what the Nao sees)
- Terminal 4:
  - Command 1: `cd sudo/brain/src` (to go the the correct directory)
  - Command 2: `./start.sh config/config_alignmain` (to execute the behaviour)

Behavior Design

Main behavior

The behavior starts from creating the subbehavior "Search", and starts the subbehavior. When the subbehavior successfully ends, it waits for 0.3 seconds for stability reasons. After the wait, it creates a new subbehavior "Approach" and starts the subbehavior. When the subbehavior successfully ends, it waits for 0.3 seconds again. Then it creates a subbehavior "Align goal" and starts the subbehavior. When the subbehavior successfully ends, it waits for 0.3 seconds before creating the new subbehavior "Kick". When "Kick" successfully ends, it will go back to the initial state and start "Search" again. Whenever any of the subbehavior fails (such as falling), it will stand up and set to initial state, hence performing the behavior from the beginning.

Search

First, the Nao stands up and raises its head so that the whole field is in the vision with its chin camera. Then, it starts moving its head counter-clockwise until it reaches angles of 0.7 radians at which point it will move its head down while the head is still in the same position on the left and move its head rightward until the head yaw is of angle 0 radians. If, the ball is in the vision during this time, then it will turn counter-clockwise with the speed of 0.3 until the ball is seen again. If the ball is not seen during the head movement, then it will turn its body clockwise with a speed of 0.3 while nodding its head up and down until the ball is in the vision. When the ball is in the vision and the ball is on the left side of the screen then it will turn with a speed of 0.1 counter-clockwise until the ball is in the center of the vision field. If it is on the right side of vision, it will turn with a speed of 0.1 clockwise until the ball is in the center of the vision field. After the ball is in the center of the vision field it will make a quick adjustment in the y-axis such that if the ball is higher up in the vision field it will look up and when it is lower then it will look down. The behavior is then finished. This subbehavior fails if the robot falls during the execution of this behavior or if it finds the ball at first but then it disappears out of sight. The ball is detected if the difference between the width and height of the blob detected is 1 or smaller. This goes for all the other subbehaviors where the ball is detected.

Approach

The head pitch angle constantly changes while it is approaching so that the ball is at the very center of the vision field. The Nao moves forward with a speed of 0.7 if the pitch angle is less than 0.2 otherwise, moves with speed 0.3 and the angle is decided based on the location of the ball in the vision field. For instance, if the ball is a little too left on the screen, it would walk while turning a little to the left until the ball is in the center of the vision field and when it is it will walk straight again (same goes for the right side). Once the ball is close enough to the robot (i.e when the head pitch angle is larger than 0.32) the Nao approaches the ball with a speed of 0.05 and after 2 s, the behavior ends successfully.If the robot falls or loses sight of the ball then the subbehavior fails.

Align Goal

First, the Nao raises its head all the way up in order to see the top bar of the goal with the chin camera. Then, it moves its head to the left until the angle of 0.9 radians and then bring it back to the initial position. If the goal is found while it is turning its head to the left, then it brings its head back to the initial position, and then it turns and starts walking counter-clockwise around the ball. If the goal is not found on during the head turning motion, then it will walk clockwise around the ball until it finds the goal. The goal is recognized if it sees the blob of the designated color with the area of at least 3500.This number was chosen because when we put the robot on the other side of the field from the goal, the size of the blob was still larger than 3500. The subbehavior fails if the robot falls.

Kick

The Nao will look down and move sideways until the ball is aligned with its left foot. Then, it will move forward slowly until the ball is in front of the left foot. Then, the Nao performs the kicking motion and successfully ends the subbehavior. This kicking motion was added to the approaching behavior by first, uploading the behavior from Choregraphe to the Nao and then getting the behavior ID (nao_kick_3-50c605) and the behavior name (behavior_1).The subbehavior fails if the robot falls or loses sight of the ball before it gets into the kicking motion.

Experimental Design

Th experiment will consist of approaching the ball at three different locations and kicking the ball towards the goal. The first location is directly in front of the Nao at 1 meter distance. The other two are located 45 degrees to the left and to the right of the Nao at 1.41 m distance. The goal is located in front of the Nao at a 2m distance. 5 trials were done for this experiment and the execution time (how long it took the Nao to align with the until it kicked) and whether or not it scored was recorded.

The time measurements were taken in a way where the stopwatch started when command 2 (ref. to Methods) was executed and it was stopped when the Nao kicked the ball.

Commit Hash

```
commit ff961561fe4cfac3e3853450b9c70158d3c94333
Author:  asd <asd@asd.asd>
Date:  Thu Mar 22 16:36:31 2018 +0100

   very final version of aligning and kicking the ball into the goal
```

Results

Table 1: Time taken for Nao to search for the ball till it aligns with the goal and kicks the ball, along with whether or not it scored

	Trial	Exec Time	Score?
Straight	1	1:09	Yes
	2	0:54	Yes
	3	1:06	Yes
	4	0:56	Yes
	5	0:57	Yes
Left 45°	1	1:16	Yes
	2	1:10	Yes
	3	1:12	Yes
	4	1:06	No
	5	1:14	Yes
Right 45°	1	1:23	Yes
	2	1:32	Yes
	3	1:27	No
	4	1:10	Yes
	5	1:12	Yes

Link to Youtube video with Nao performing the goal alignment and kicking (1 trial from each position): https://www.youtube.com/watch?v=JAGHJ7QZBJE

Conclusion

The robot was able to find, approach, and kick the ball towards the goal in a timely manner (Straight: 1 min, Left 45°: 1 min 11 sec, Right 45°: 1 min 20 sec, Average of the 3 angles: 1 min 10 sec). The kick was accurate and scored with a high rate (Straight : $\frac{5}{5}$, Left 45°: $\frac{4}{5}$, Right 45°: $\frac{4}{5}$, Total of the 3 angles: $\frac{13}{15}$).

Discussion

The biggest issue we had during the experiment was nothing to do with the code, but the calibration. The brightness of the room constantly changed by light sources from outside, and the lights in the room created shadows which became problematic when the ball was in the shadow of the robot. We fixed the issues of the shadow by changing the placing of the goal.

The 2 shots that the robot missed was caused by the last approach to the ball before the kick. After the robot aligned itself with the goal, it looked down and tried to adjust so that the ball was in front of its left foot. However, when the robot tried to adjust the x-axis placement, we noticed that it was slightly turning clockwise. We solved this problem by making the robot slightly turn counter-clockwise instead so it is straight again. This solution, however, may increase in deviation the more the robot has to move for the adjustment. The better solution would be to know where the goal is while the robot moves towards the ball, preferably using both the chin and the head camera.