Team Thirteen Paper

Personal note: MASlab has been an interesting and enjoyable use of 4 weeks. As a Harvard student, most of my classes are completely theoretical and I appreciated the chance to actually *build* something. Although I never managed to register for the class, and was taking it during Harvard's period of finishing final papers, projects and taking final exams, I was surprised at the amount of time I wanted to put into the class. When Finale first said that 8ish hours a day were expected for each person at the begininng of the course, I was skeptical. In the end however, I wished I could have vested even more time. Speaking of Finale, she did a wonderful job leading the course and I really appreciate all of her help. I also appreciate all of the help the staff gave us. I am really thankful for all the time all the members of the staff put in, we wouldn't have managed to finish anything without it. **-Ryan Carlon**

And now on to the paper

As with every other team, ours had grand designs at the beginning of the competition. Not knowing many of the details till the first day of class, we didn't start seriously discussing our options till the beginning of that week. Our most aggressive of ideas, and one that we were seriously considering implementing until well into the second week, was to float a second webcam above our robot using a helium balloon. The combination of ground and overhead views would make finding balls, goals, walls and paths to balls an easy task. This venture failed when we realized the helium balloon was not strong enough to lift the webcam. Other fun ideas came to mind too, especially flinging balls through the field goal, as this method of scoring was all but begging for a violent and entertaining solution. Grand mapping schemes were also bandied around, including rotating our IR sensor, using Kalman filtering and using the bar codes as an accurate reference point.

Practicality soon presided however, as it became apparent that we would be hard pressed to finish the robot in time with anything that wasn't absolutely necessary. Our overall strategy needed little discussion, as scoring field goals was really the only way to go point-wise. Our initial goals were to complete the frequent checkpoints and daily goals presented by the staff during the first two weeks. We had some ideas for our overall design, but nothing concrete until the second week. As the the deadline for Checkpoint 2 came and passed, we had started construction on our robot, including the pincher and the shaft encoders, however the majority of the body wasn't complete for the next couple of days and we weren't able to pass Checkpoint 2 until Mock Contest 1. By Mock Contest 1, the color detection code was finished, including fixing all the HSV issues that presented themselves. Our barcode recognition was also complete by this point, with most of the code to this point credited to Olivier. Antoine and Renaud had also finished the bin for storing robots by this point and our robot was basically finished, only small modifications happened after this. Soon after Olivier and Antoine left, Renaud and Ryan realized that the main focuses needed to be on wandering code and scoring. Mapping could be implemented only if this worked quite well. We spent the rest of our time improving our wandering code and making sure we caught corner cases. We did end up implementing mapping, although only using dead reckoning and our gyro as we ran out of time to do anything more interesting.

Our mechanical design ended up being one of the simplest of the competition. Renaud and Antoine thought that a pincher would be the best method of lifting balls to the top of the robot and quickly designed a very effective one. While most other groups skipped a pinching mechanism citing the need to be very precise and likelihood of failure when attempting to pick up balls, our system ended up being one of the most effective and reliable. It was able to lift up to two balls at once, nearly almost grabbed any ball it lined itself up in front of, and was even able to grab balls stuck in very narrow corners (after several tries). The key feature of the pincher was the wide dual leveled pinchers. The wideness allowed us to be less precise when lining up with the ball, and the dual levels secured the ball well, even when the pincher swung upside down to deposit the ball in the storage bin. Driving the pincher were two servos, as we wanted to be able to direct the movement of the arm fairly precisely. One servo closed all four of the pincing arms, which were connected using bolts and gears, and the other servo controlled the rotation of the pincher's arm. After Mock Contest 2, we realized that our pincher sometimes got bent if it ran into a wall or attempted to pick up multiple balls and Renaud fixed this by adding springs to the ends of the 3 of the 4 arms. The last arm was attached directly to the servo and so did not bend out of place. Once we had a way to lift balls, we needed a place to store them and a way to score them. The most obvious solution was a dumpbin, and once again Renaud and Antoine went to work. Our final design was a flat piece of wood which pivoted back and forth at the center of the robot. At the front corner of the bin was half a piece of pvc pipe which provided a tube for the balls to roll out into the goal. The bin usually started tipped backward, so the balls would stay there while the robot drove around. There was a constant pressure applied to the front of the bin by a rubber band so when the servo holding the back of the bin down released it, the bin would tip forward and dump the balls out. While occasionally balls would become jammed, by using the code to tip the bin several times, the jam was always dislodged and the balls in the bin always scored. It was difficult to line up the tube without hitting the field goal uprights as the tube was off center and as evidenced in the competition, we never found a convincing solution to this problem. The chassis of our robot was built foremost to accomodate securely all the various components. We decided on the pincher and dump-bin first, then built the chassis around those and the computer. Having limited machine shop access, wood was the ideal material for us to build from. We also wanted a design that would get stuck as little as possible. These factors resulted in our final design, which had a curved rear-end, a flat front end, and recessed wheel spaces on the sides of the robot.

Having spent half of our sensor points on servos for the arm and dump-bin, we were somewhat limited in navigational and mapping sensors. After seeing the shaft encoders provided to us, we realized that these would not be nearly as accurate as we wanted. Renaud and Antoine had the brilliant idea of building home-made shaft encoders using the ones found to track the ball in a computer mouse. After some time spent building them, including building an additional circuit to create an accurate reading (the voltage supplied internally by the orcpad complicated things), the shaft encoders reliably worked. The final versions registered over 700 ticks in a revolution, letting us know our position quite accurately. We also wanted a more accurate representation of our angle, and so used the provided gyroscope. Although its tendancy to drift with time was a concern, we were able to use it effectively by resetting frequently and using a long calibration time after the gyro was initialized. Using the shaft encoders along with the gyro allowed us to know where we were as well as an approximate location of where we wanted to go.

In order to actually wander, we needed sensors to determine the location of walls and corridors in front of our robot. To do this, we used one long range IR sensor placed near the center of the robot, and 2 medium range IR sensors placed on each side. Each of the two side IR sensors had 2 states, seeing a wall a bit away, and seeing a wall close up. If the sensor saw the wall a bit away it would cause the robot to turn away slighty, and if it saw a wall close up, the robot would make a more dramatic turn. This allowed our robot to move pretty smoothly along corridors, and at the same time not get stuck that frequently. The long range sensor was used to find corridors to go to. As the robot turned, the IR sensor would find discontinuities in the wall, which corresponded to branching paths. This seemed to work only turning in one direction to the discontinuity, for a reason we never discovered. In order to keep the robot from spinning in circles, we also set the IR sensor to report any distances greater than 1.5 meters, which the robot would then venture toward.

Software design was where things became more complicated. We decided to create classes to keep track of all the objects we saw on the camera, as Java does this quite well. We scanned through the image pixel by pixel, marking and coloring each as its object type. In order to single out collective objects, the locations of key pixels (like smallest x value, last x value, highest y value....) were stored and compared against. This code was quite complicated and not nearly as clean as it could have been, yet worked - most of the time - extremely well. We also spent a good deal of time working on identifying barcodes, and were able to give how many barcodes were found and the value of each, although this code was unfortunately not used for the contest. This part of the code also included methods for determining the location of the ball with the most number of pixels (most likely the closest ball) and the location of goals. These methods were used extensively in the rest of our code. This image scanning code was threaded as well, so it was nearly always running. The wandering bit of code went through a couple of iterations before we decided to use wall following. To this end we had the robot always turning slightly left. The basic structure of our code was a finite state machine, where it would wander around unless it saw a ball or a goal (while currently holding a ball). The robot would travel left-straight along the wall till it ran into a corner, then would spin looking for a way out. If it found a ball or goal, it would go toward it, avoiding walls in the way. This aspect was particularily robust, even in somewhat nasty corridors.

After finishing a consistent and effective wandering code, we had just enough time to develop a very simple mapping algorithm. Using our wandering code, we took out the goal scoring and ball catching methods, and instead added methods to store locations where balls and goals were found using our absolute location determined by the shaft encoders. We also added an 'interesting area' method into the mapping code, which prevented us from mapping the same balls/goals twices, although as evidenced in the final competition, this could have used some fine tuning. At the end of the mapping code, a method was added to select the top three locations with the most balls and discard the rest. The wandering code then went after the areas in order of number of balls, giving up if the location wasn't reached in thirty seconds. After all the mapped locations were reached or given up on, the robot went back to random wandering. The robot would also use the mapped data to try and find a goal it had mapped if there were less than thirty seconds in the round and it was carrying a ball.

We also had two programs that allowed user interface to the bot by using the keyboard and botclient. One of these programs was a trivial drive method that allowed the user to remotely drive the bot. The other was much more helpful and changed the levels of red and blue in the picture until the bot recognized the elements of the playing field in the current lighting conditions.

Overall our robot did a pretty good job. It needed to be speeded up, as the course was too spacious for the bot to adequately cover in the allotted time. While our robot would have done well in narrow, zig-zagging corridors, it was somewhat overwhelmed with the space presented to it. Otherwise our robot was quite consistent and effective. Had we had a bit more time, robust mapping code could have been developed and our robot could have been amazing. Alas.

One of the key points learned in the class is that simple is good. Our robot did better than most other robots because it was pretty much in its final configuration by the first mock contest. By being able to devote a lot of time to develop code without having to worry about changes in bot design, our final code was quite reliable and consistent. If it were to be done again, we would have started earlier and had a working prototype much earlier in the month. Time flew by much quicker than originally anticipated. Overall we were pleased with the progress we made, and only wish there had been more time to implement even niftier things.

Team Poutrelle
