Sheep-dog Problem

Rajiv Mantena[1], Bhanu Kiran Chaluvadi[1], Clinton Fernandes[1], Kameswari Devi Ayyagari[2]

[1]Department of Mechanical Engineering and [2]School of Computing, University of Utah

December 10, 2015

Abstract

In this paper, we try to explore the problem space of planning the motion of a dynamic body by influencing it with another dynamic body. To explore the problem space, we apply this to the sheep dog problem where we program the robotic dog to herd the sheep towards the goal.

1 Introduction

Motion planning plays a major part in the today's robotics scenario. Although, there has been a lot of work done in planning the motion of a body between initial state and the goal state, the problem space of planning the motion of a body relative to another body has been relatively unexplored.

We are trying to address the problem of moving an object(or a set of objects) to a desired goal state by influencing the object(or a set of objects) to be moved using another object that already has a notion of where the goal is.

1.1 Definitions

- Driver: The object or set of objects that are acting as a driving force. They influence the other objects in the environment by exerting either an attractive or a repulsive force on the other set of objects in the environment.
- Driven: The object or set of objects that are being influenced by the driver. They are either attracted or repulsed by the driver and driven towards the goal.
- Environment: The space where the driver and driven exist.
- Initial State: The initial state(position, velocity, angle of orientation) of the driven in the environment.
- Goal State: The state(position, angle, velocity) that the driven needs to be directed to.

1.2 Assumptions

- All the objects(driver, driven and obstacles) follow the laws of physics.
- The current state of every object in the environment at every instant in time are available or can be computed.
- The driver has a sense of the goal and is capable of exerting an influential(either attractive or repulsive) force on the driven.

We attempted to solve the problem of planning the motion for the driven by exerting an influential force on the driven by the driver by solving the sheep-dog problem which will be discussed below.

1.3 Sheep-Dog problem

Let us consider a flock of sheep in a grazing field. The sheep graze around in the field. Let us consider the problem of planning a robotic dog that would herd the flock of sheep towards the goal. The sheep would be scared of dog. So,the sheep will be repulsed by the dog, moving away from the dog. We assume that the dog has a sense of where the goal is and, the dog while repulsing the sheep away from itself also moves the sheep towards the goal.

The sheep in the above problem are the driven, the dog the driver and the grazing field, the environment. In this paper, we try to plan the motion for a robotic dog that would herd sheep towards the goal. To solve this problem, we use the concept of potential fields [2].

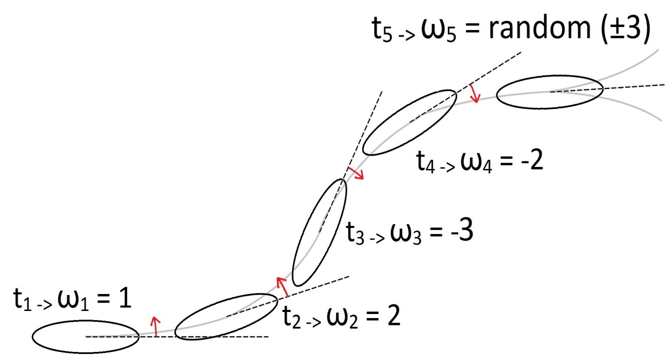

Fig 1: Randomness of dynamic objects

2 Motivation

2.1 Potential Field Theory

Every stationary body has potential energy associated with it. Force is a derivative of the potential energy. The potential field theory works with the concept that when a robot approaches the stationary body in the environment, the stationary body exerts a force that could either be repulsive or attractive in nature on a robot. The magnitude of the force exerted on the robot depends on the distance between the object and the robot. The farther the robot is from the object, the weaker the force exerted on the robot.

In the approach that we use, we intended to use the potential field theory to achieve the replusive force between the dynamic objects, rather than using it to plan the motion of the objects. Traditionally, potential fields have been used to plan the motion of objects by implimenting the concept of objects tending to move from higher potential positions to lower potential positions. We, however didn't follow this approach. As dynamic objects have kinematic energy, they apply a force either repulsive or attractive on other objects. In our context, the driver exerts a repulsive force on the bodies being driven, driving them away from the driver. We use this phenomenon to use the dog to herd the flock of sheep away from the dog and towards the goal. This can be represented by the below equation.

$$F_{ds} \propto d_{ds}$$

Where, F_{ds} represents the repulsive force applied by the dog on the sheep and d_{ds} is the distance between the dog and the sheep.

3 Methods

We decided to initially implement this in 2D, and then port it into 3D. We decided to use Box2D [1] as the simulation environment. Box2D is an open source C++ physics engine that simulates rigid bodies. The dog and the sheep are represented as 2-Dimensional objects in the Box2D environment. (Note: Defend why Box2D)

3.1 Environment Setup

- A dynamic body in the Box2D environment to represent a dog
- Five dynamic bodies in the Box2D environment to represent sheep.
- Static bodies that represent boundaries and obstacles (if any).
- Random motion is assigned to the sheep in the environment. At any instant in time, for the

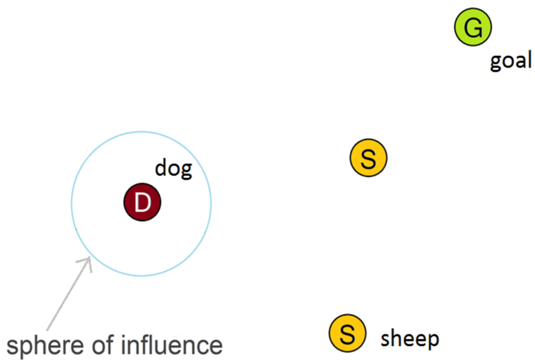

Fig 2(a): Dog, Sheep and Goal at initial positions

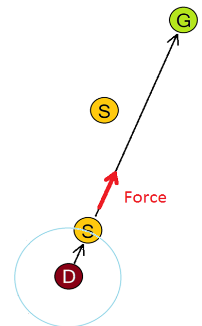

Fig 2(c): Dog moves forward which applies a repulsive force on sheep towards the goal, until it finds another sheep which is farther than current sheep

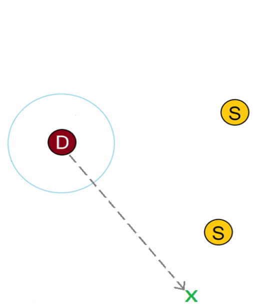

Fig 2(b): Dog aligns itself behind the farthest sheep, ensuring the sheep doesn't fall within its sphere of influence

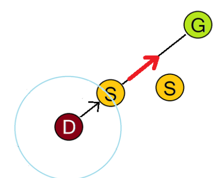

Fig 2(d): Dog executes the same set of steps on the new sheep object

sheep, the randomness is defined only in the forward direction as sheep do not move sidewards or backwards without first turning in that particular direction.

To achieve this non-holonomic motion [3] [4], we have assumed that the velocity of the objects stays fixed unless it collides with something and the randomness is taken care of by varying the angular velocity. We have assumed the dog's and sheep's speeds to be 3 units per timestep and 1 unit per timestep respectively. All the experiments have been performed maintaining these to be constant. A random value would be applied to be the angular velocity of the object, ω. This randomness is defined by the below function.

$$\omega = (rand()\%6) - 3$$

We used this random function to associate necessary randomness for the objects. The angular velocity is randomly assigned a value between -3 to +3 $radians/sec^2$. Figure 1 explains how the object motion is obtained based on this function. The figure depicts the path of an object after 4 timesteps. The region between the gray splines after the object at t_5 shows the possible position of the object at the next timestep.

- The dog is introduced in the environment at a random position.

The dog at every time step, finds the sheep that is farthest from the goal, aligns itself behind the sheep such that, the dog, sheep and goal are collinear. At this point, the sheep is still outside the dog's sphere of influence. Figure 2(a) and 2(b) represent this sequence of actions. The dog now starts to proceed towards the goal. Once the sheep falls within the sphere of influence, a repulsive force is applied on the sheep. As the dog, sheep and goal are all collinear, the repulsive force is always towards the goal. The same is depicted in Figure 2(c). The dog will proceed towards the goal until another sheep is farther than the current one. If the above condition occurs, the dog aligns behind it, similar to how it had aligned behind the previous sheep. Figure 2(d) represents how the dog aligns and applies a repulsive force on current sheep. The process repeats until all the sheep have reached the goal.

Algorithm 1 Algorithm for herding the sheep without heuristics

```
pos = Position of the Dog
if pos is valid then
    FS = Farthest sheep from the goal
    Align FS towards the goal
    Apply repulsive force on FS
end if
for s =0 to number of sheep do
    if S is not FS then
        Continue grazing
    end if
end for
```

We optimized the algorithm further by introducing the following heuristics to the algorithm.

- HG : Distance between each sheep and the goal.
- HD : Distance between each sheep and the dog.
- HV : Velocity heuristic for each sheep.

We also introduced a scaling factor to each of the heuristics. In the optimized version of the algorithm, the final heuristic with the weights added, is calculated for each sheep. The dog picks the sheep with the largest heuristic, aligns itself behind the sheep and applies a repulsive force on the sheep towards the goal. This is outlined in Algorithm 2. By varying the scaling factors for each heuristic, we conducted experiments to analyse what scaling factor applied to which heuristic would result in optimum results. The results and experiments will be discussed in the sections below.

Algorithm 2 Algorithm for herding the sheep with heuristics

```
for s =0 to number of sheep do
    HD = distance between s and the dog
    HG = distance between s and the goal
    HV = Velocity of sheep
    FH = SF1 * HD + SF2 * HG + SF3 * HV
end for
pos = Position of the Dog
if pos is valid then
    FS = Find the sheep with the highest FH
    Align FS towards the goal
    Apply repulsive force on FS
end if
for s =0 to number of sheep do
    if S is not FS then
        Continue grazing
    end if
end for
```

4 Experimental Analysis

During our initial analysis, we have recorded the number of iterations executed to successfully gather the sheep into the pen and the time taken from begin to end. It has been observed that the time taken and the number of iterations performed are linearly proportional to each other. As each iteration executes at a specific timestep, this relationship is explained. Hence, we have taken into account only the time executed as it provides a much better representation for intuitive understanding. All the times mentioned below are average of 3 individual times from independent experimental executions. We have noticed significant deviation in execution times for the same

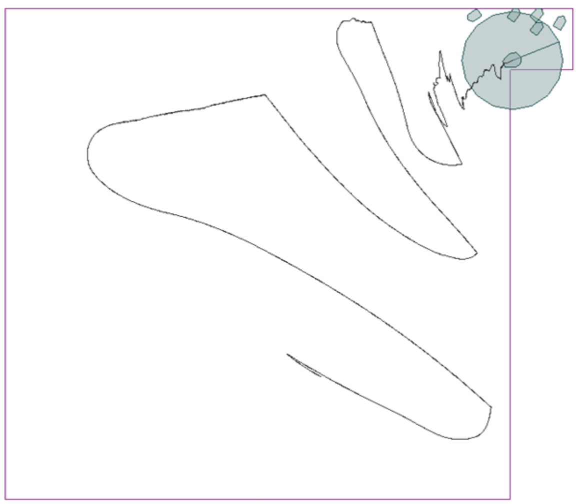

Fig 3: Path traced by the dog based on Algorithm 1

case due to the presence of randomness in the positioning and the motion of sheep.

From our experimental results performed with Algorithm 1, we noticed an average execution time of 54 seconds (approx. 3231 iterations). The algorithm was successful 9 out of 10 times. It failed once in 10 times whenever a sheep stays in contact with an obstacle or boundary and the dog tries to apply a force further towards the static objects, resulting in a locked situation of the sheep. Figure 3 shows the path traced by the dog using Algorithm 1. It could be noted that the distance is proportional to the time taken for execution, as the objects trace only a fixed distance in a timestep.

However, the findings from execution of Algorithm 2 have been much more optimistic, with certain executions taking as less as 17 seconds. For initial tests, we have chosen random weightages for each of the three heuristics : HG being 0.5, HV being 0.05 and HD being 0.5. With these weightages, we were able to reduce the average execution time to 34.51 seconds. To further tune the weights of the heuristics, we followed an approach which is explained below.

Firstly, we have tuned HG and HD keeping the HV constant. We chose HD and HG as per the below conditions.

$$0 < HG < 1$$
$$0 < HD < 1$$
$$HG = 1 - HD$$
$$HV = 0.05$$

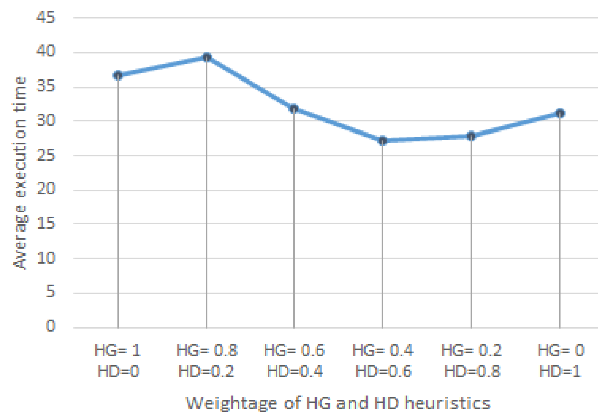

Fig 4: Graph showing the trend of execution time vs weights of HD and HG

The average execution time has been measured with varying HG from 0 to 1 in steps of 0.2. Figure 4 represents the findings. We observed the least average times for HG = 0.4 and HD = 0.6. These values have been saved for further tests, as they were the most optimal scaling factors for HG and HD..

Later, once we have obtained the most efficient scaling factors for HG and HD heuristics, we varied HV while keeping HG = 0.4 and HD = 0.6. The trend of the average execution times is represented in Figure 5. We have observed the least average times at HV = 0.25. Hence, we noted the most suitable scaling factor for HV to be 0.25.

With the above identified optimal scaling factor values, experiments have been repeated with Algorithm 2 and we have observed the average execution time to have reduced to 27.14 seconds. Figure 6 illustrates the path traced by the dog by using the Algorithm 2. Comparing Figure 3 and Figure 6, we infer that the path that the dog traced using Algorithm 2 is much shorter than the path traced using Algorithm 1.

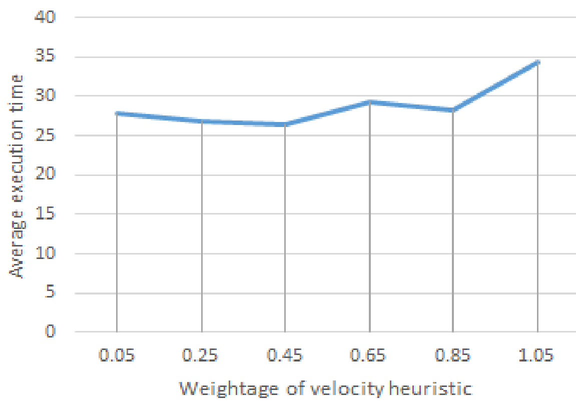

Fig 5: Graph showing the trend of execution time vs weight of HV

Fig 6: Path traced by the dog based on Algorithm 2

5 Future Work

During our implementation, we noticed that there are several optimizations that could be further applied to our algorithm.

- We think that introducing multiple dogs at strategic positions in the environment would considerably reduce the time taken by the drivers(dogs) to herd the driven objects(sheep). For instance, the dogs could be placed at the four different corners in the environment and each dog can herd the sheep in it's own quadrant. Since the dogs would be working together simultaneoulsy to push the sheep that are local to them, we think that the overall distance travelled by the dogs to herd the sheep and the time taken for the sheep to be herded will be reduced.

- The sheep have a tendency to move in flocks. So, a force of attraction between the sheep within particular range of each other could be introduced. We haven't currently accounted for the force of attraction between the sheep. We think that this would also considerably reduce the time needed to herd the flock of sheep.

- In our current approach we are only taking into account the current position of the sheep. We could also predict the position of the sheep in the future and plan the motion of the dog based on the future position of the sheep. We think that proactively finding the positon of the sheep and planning the motion of the dog accordingly would considerably reduce the taken to herd the sheep towards the goal.

- In our current approach, we take into account the distance of each sheep to the goal. We could compute the sum of the distance between each sheep and the goal and plan the motion of the dog in such a way that the total distance travelled by the sheep is reduced. We think that it would be interesting to see how the addition of this heuristic would impact the total time taken to herd the flock to the goal.

6 Conclusion

We could successfully direct the sheep towards the goal based on the motion of the dog. We think that exploring the field of planning motion of a dynamic object governed by another dynamic object seems exciting and feasible.

We would like to thank Tucker Hermans for all the guidance and help provided during the course of the project.

References

[1] C++ based physics engine-box2d. `http://box2d.org/`.

[2] Potential fields for real time simulation. `http://aigamedev.com/open/tutorials/potential-fields/`.

[3] Jiadong Li, Shirong Liu, B. Z. X. Z. Rrt-a* motion planning algorithm for non-holonomic mobile robot. SICE Annual Conference 2014.

[4] R. Zulli, R. Fierro, G. C., and Lewis, F. Motion planning and control for non-holonomic mobile robots.