What's Up, Dock?
Provably Safe Boat Maneuvers

William Ganucheau (wganuche)

May 9, 2017

Abstract

From shipping, to leisure, to military, boats are critical to society, and with any prevalent system, proofs of safety are a must. In this paper, I propose a simplified model of surface vehicles and provide two safe controllers for the model. The first controller defines allowable steering commands in order to respect a maximum linear and centripetal acceleration while remaining within a rectangular bounding region. The second controller provides an upperbound for the distance between a boat and a docking location such that the boat will not collide with the dock if it cut its engines at its current location with its current velocity. I show that this particular controller is between 40-70% as efficient as an optimal controller over a reasonable range of docking velocities.

1 Introduction

Over ninety percent of all shipping happens via cargo ship [3], cruise ships carry over 20 million passengers every year [1], and the U.S. Navy has 275 deployable battle ships [4]. There is no doubt that boats are pervasive in modern society. However, our reliance on the technology is also a liability. At best, an accident involving one of these vessels costs hundreds of thousands of dollars, and at worst, human lives. For this reason, we want to be sure that we have formal guarantees about the safety of these vessels. In developing such models, we also pave the way for automation, increasing efficiency and allowing humans to perform more interesting, and potentially safer tasks.

The problem of designing controllers for boats specifically is interesting for variety of reasons. For example:

- Unlike cars which have to remain on roads, Boats spend most of their time in open environments. This means there are many more valid steering options for a given state. In this project, we attempt to prove that as many of them are safe as possible.

- Traveling through water means that we can't just ignore drag as we might for a land vehicle. The drag of the water fundamentally changes how we control the vessel. For this model in particular, we assume that the boat doesn't have any form of active breaking. That is, the only way for the boat to slow down is to reduce (or cut) its thrusters and let the drag do the rest of the work.

In this paper I first provide a model for maneuvering within a safe region while maintaining a safe linear and centripetal acceleration. Then I provide a controller capable of docking safely provided the distance between the boat and the dock is above a certain threshold. I'll show that this distance threshold is within 40-70% of the optimal threshold for a particular range of initial velocities.

2 Related Work

In [5], the authors developed a model and simulation for an unmanned surface vehicle capable of maintaining a specific heading. While they achieve favorable results, the simulated environment was boundless and hazardless and the specific task performed was quite simple. Further, their autopilot was based on heuristics rather than a proof of correctness. My work aims to supplement this work by showing that a provably correct control scheme exists for more complicated tasks and environments.

Unlike [2] which uses computer vision and signal processing techninques to accurately dock an underwater vehicle despite noisy data, but has no proof of correctness, I assume perfect sensor information (location of the vessel relative to the dock) and provide a provably safe docking strategy.

3 Model

It is important to choose a model that is simple enough that we can prove things about the model, but also close enough to reality such that our results are meaningful. In this project, the steering capability of the boats was such that they could move in arbitrary circular paths. The steering capability modelled is suitable for a wide variety of real-world boat configurations: Dual rear thrusters, single thruster and rudder, or azimuth thruster(s). A complete description of the system state and defining constants is shown in Figure 1.

Variable	**Description**
x	X position of vehicle
y	Y position of vehicle
m	Thrust generated by the motor(s)
r	Radius of circle currently being travelled
Constant	**Description**
C_d	Drag coefficient
A	Wetted area
R_{min}	Minimum radius achievable
M_{max}	Maximum thrust achievable

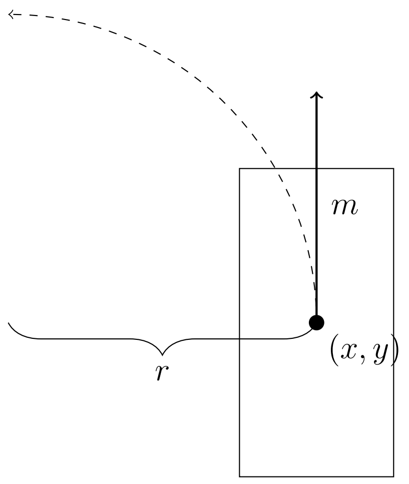

Figure 1: Overview of system state and constants[1]

Physics are a Drag The physics of the system were highly simplified for modelling purposes. First, I assumed that the boats were never moving fast enough to hydroplane and were therefore always experiencing a drag force from the water. Additionally, I assumed that the drag due to air resistance was negligible and didn't account for changes in wetted area due to pitching or rolling. The main two forces involved with the motion of the boat are the thrust, m, and the drag force which is given by

$$F_d = -\frac{1}{2}\rho C_d A v^2$$

[1]In reality, A would be a function of our current radius of travel. However, what's important is that A is contstant during the differential evolution of the system. One could substitute a more complicated calculation of A and the proof would work out exactly the same.

For our purposes, the variable C_d is assumed to absorb the $\frac{1}{2}\rho$ factor as well.

Circular Motion Circular motion with radius r and linear velocity v can be described as

$$\begin{aligned}
x &= r\cos\left(-\frac{v}{r}t\right) \\
y &= r\sin\left(-\frac{v}{r}t\right) \\
&\Rightarrow \\
x' &= v\sin\left(-\frac{v}{r}t\right) \\
&= \frac{vy}{r} \\
y' &= -v\cos\left(-\frac{v}{r}t\right) \\
&= \frac{-vx}{r}
\end{aligned}$$

Modeling Dynamics Concretely, the following dynamics were used:

$$\begin{aligned}
d_x' &= \frac{vd_y}{r} \\
d_y' &= -\frac{vd_y}{r} \\
x' &= vd_x \\
y' &= vd_y \\
v' &= m - C_dAv^2
\end{aligned}$$

Here d_x and d_y represent components of the unit vector in our current direction of travel. For the purposes of making the model as realistic as possible, all the controllers were *time-triggered*. In other words, we guaranteed that the discrete controls would run at least once every T seconds, but no other guarantees were made about how long the continuous dynamics were allowed to evolve.

Reality Check Those who have ever been in a boat moving at a reasonably high speed should immediately become suspicious of the proposed dynamics. According to the preceeding dynamics, as soon as a new steering commmand is issued, the boat will *immediately* begin perfectly following the commanded arc. Even assuming perfectly calm waters, this is not accurate. In reality, the boat will have some "drift" as it goes from one circle to another, especially if the change is drastic. Modeling this, however, makes the problem much more difficult. However, that's not to say that this model is useless. The aforementioned "drift" is minimized in reality at low speeds or by not making such drastic changes. Therefore, an implementation of this controller should limit the change in r and m between timestamps to be relatively small or only maintain speeds below some threshold.

4 Driving

The first model I proved demonstrated that 3 essential safety properties could be maintained while driving the boat:

1. The boat will remain inside some predefined, static "safe" region.
2. The boat will never obtain a linear acceleration with magnitude greater than some fixed limit $A_{\max}$.
3. The boat will never obtain a centripetal acceleration with magnitude greater than some fixed limit $C_{\max}$.

For this problem, the controller used is very simple. We choose a radius and a thrust, and then ensure that it satisfies the conditions described in this section:

```
r :=*; ?(r^2 >= rMin^2);
th :=*; ?(th >= 0 & th <= thMax);
```

4.1 Note on Nondeterminism

In the controller above, we choose r and m (`th`) nondeterministically. The proof described below will then then "check" that they satisfy some properties that allow us to conclude the system in safe. This is obviously not how you would do things in practice; one would have a deterministic way of assigning these values such that they are guaranteed to satisfy the given constraints. The advantage of including non-determinism in our models is that it allows us to prove safety in the broadest sense possible. We know that if a set of steering commands is safe, we've proved it, and vice versa. We're not artificially limiting our commands to a more restrictive subset than what is necessary to prove the desired properties.

4.2 Respecting Boundaries

To begin with, I defined the "safe zone" as a simple rectangle. Assuming the boat starts within the rectangle, it's easy to remain within the rectangle by only choosing radii such that the circle defined by that radius stays entirely within the circle as seen in Figure 2. This is slightly conservative; just because part of our circle goes out of bounds doesn't mean we'll actually drive out of bounds. We could have the boat stop before reaching the boundary and then steer away from it. Practically, though, this simplification doesn't severely limit our steering decisions but it greatly simplifies the proof. I also showed that we can define arbitrarily complex safe regions that are the union of overlapping rectangles. The logic is identical: only allow circles that are entirely within at least one of the safe regions. We can switch between rectangles by driving along a circle that is completely within multiple safe regions, as illustrated in Figure 3. The addition of these "super regions" can also serve as a means of static obstacle avoidance. By surrounding each obstacle with a sufficiently large bounding box, one can then decompose the remaining space into a set of overlapping regions, none of which contain an obstacle.

Proof Putline Proving this property is fairly straightforward. To begin, we can compute the center of the circle we're travelling along for any choice of r:

$$(c_x, c_y) = (x + rd_y, y - rd_x)$$

Then, if we define our safe region by $(x_{\min}, x_{\max}, y_{\min}, y_{\max})$ we check/test that this circle is completely within the desired region:

$$\begin{aligned} c_x + |r| &\leq x_{\max} \wedge \\ c_x - |r| &\geq x_{\min} \wedge \\ c_y + |r| &\leq x_{\max} \wedge \\ c_y - |r| &\geq x_{\min} \end{aligned}$$

Leveraging the fact that we can prove we are always on the circle:

$$(x - c_x)^2 + (y - c_y)^2 = r^2$$

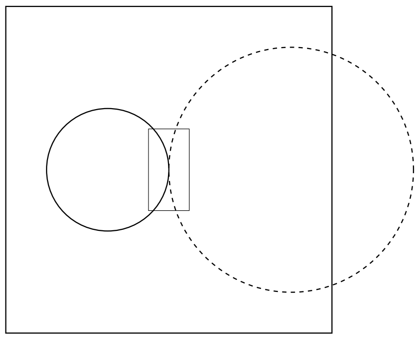
Figure 2: Example of a valid radius (solid) vs. an invalid one (dashed)

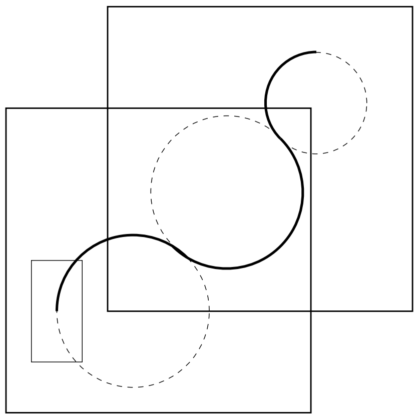
Figure 3: Example of manuevering between regions in a complex environment

We can then conclude that the following postcondition is always satisfied:

$$x_{\min} \leq x \leq x_{\max} \land$$
$$y_{\min} \leq y \leq y_{\max}$$

Adding multiple regions doesn't change the proof at all. We just check if the circle is in *any* of the regions, and likewise conclude at the end that we're in at least one of them.

4.3 Limiting Acceleration

4.3.1 Linear Acceleration

At each timestep, we non-deterministically choose a new thrust, and we want to ensure that this thrust will not cause the acceleration to exceed $A_{\max}$. The acceleration of the boat is given by

$$a = m - C_d A v^2$$

Given infinite time, this means that the velocity we're travelling at will reach a steady state at

$$v_{\text{term}} = \sqrt{\frac{m}{C_d A}}$$

Note that the magnitude of the acceleration is monotonically decreasing:

- If $v < v_{\text{term}}$, then we will speed up. The boat will experience a large initial acceleration that will decrease as v approaches v_{term}
- If $v > v_{\text{term}}$, then we will slow down. The boat will experience a large initial decceleration that will subside as v approaches v_{term}
- Finally, if $v = v_{\text{term}}$ then no acceleration will be expereinced and the boat will continue travelling at the same speed

Therefore, we simply need to ensure that the initial acceleration has a magnitude less than the maximum allowed acceleration:

$$(m - C_d A v^2)^2 \leq A_{\text{max}}^2$$

Proving Asymptotic Behavior The informal argument above may be convincing to a human, but something more formal is required by KeyMaeraX. Normally, when we want to prove that something is true over the course of the evolution of the dynamics of the system, we use the `dI` rule, which stands for differential induction. For example, if we want to prove that $p < q$ at all times, it would suffice to show that $p < q$ holds initially and that $p' < q'$ (where p' refers to the derivative of p). This argument essentially says that the thing we're trying to prove gets *more* true over time: q always grows at least as fast as p. Now, consider the case that $v < v_{\text{term}}$. Observe that this property actually gets *less* true over time–v gets closer to v_{term}, not further. One way of approaching proofs of this type in KeyMaeraX is to use a differential ghost which introduces a new variable to prove a property of interest. For example, if we can find some $b > 0$ such that $b^2 p = -1$, is a differential invariant, we can then prove that $p < 0$ is a differential invariant of the system. In particular:

$$\begin{aligned} \exists b > 0.\ b^2(v - v_{\text{term}}) = -1 &\Rightarrow v - v_{\text{term}} < 0 \\ &\Rightarrow v < v_{\text{term}} \end{aligned}$$

It turns out one such b is

$$\begin{aligned} b &= \sqrt{\frac{-1}{v - v_{\text{term}}}} \\ b' &= C_d A v \end{aligned}$$

The same argument works for proving $v > v_{\text{term}}$.

4.3.2 Centripetal Acceleration

The centripetal acceleration experienced by the vessel is given by

$$A_c = \frac{v^2}{|r|}$$

For safety, we wish to prove that this expression is always less than some maximum, C_{max}. Proving that this holds over the course of the differential evolution of the systems relies on almost identical reasoning to that above. Since we've proven that at all times $v_0 \leq v \leq v_{\text{term}}$ (or vice versa if $v_0 \geq v_{\text{term}}$) we simply need to check that $\frac{v_0^2}{|r|} \leq C_{\text{max}} \wedge \frac{v_{\text{term}}^2}{|r|} \leq C_{\text{max}}$ in order to know that this property holds over time.

5 Docking

5.1 Problem Definition

When proving the safety of docking, the question we want to answer is "How long can I wait before cutting my engines without running into the dock?" Specifically, we assume the boat is starting at $x = 0$ with $v = v_0$ and travelling in a straight line towards a dock located at $x = x_{\text{dock}}$. The boat immediately cuts its engine ($m = 0$), and we'd like to determine how low the value for x_{dock} can be while still being sure we won't run into the dock (defined by $x > x_{\text{dock}}$). One minor caveat is that, using the dynamics described in Section 3, the boat will actually never reach a velocity of 0, it will only get asymptotically close. Therefore, we choose a velocity v_{stop} such that if the boat reaches a velocity less than v_{stop}, we consider the boat as being "stopped". We enforce this in the model by adding $v \geq v_{\text{stop}}$ to the evolution domain of the differential equations. So the goal is to reach this velocity before we reach the dock. Due to the nature in which we've defined the problem, the controller for this model is extremely trivial: cut the motors, and let physics do the rest.

5.2 Exact Solution

Let's begin by looking at what actually happens in nature. Observe that the differential equation $v' = -C_d A v^2$ (which is our acceleration one we've cut the thrusters) has the following as a solution:

$$v(t) = \frac{v_0}{C_d A v_0 t + 1} \tag{1}$$

Which, when integrated, gives the following solution for x:

$$x(t) = \frac{\ln\left(C_d A v_0 t + 1\right)}{C_d A} \tag{2}$$

Now, if we solve (1) for t, we can get an equation for how long after cutting our engines it will take for us to reach any given velocity. In particular, we're interested in knowing how long it will take for us to reach v_{stop}:

$$t = \frac{v_0 - v_{\text{stop}}}{C_d A v_0 v_{\text{stop}}} \tag{3}$$

Finally, we can plug in (3) into (2) to determine exactly how far we will travel before reaching the threshold velocity:

$$\Delta x = \frac{\ln\left(\frac{v_0}{v_{\text{stop}}}\right)}{C_d A} \tag{4}$$

So in reality, as long as x_{dock} is greater than this expression, we would be safe. Unfortunately, KeyMaeraX doesn't know about ln. Therefore, we have to find a polynomial expression that upperbounds Δx.

5.3 Proof

One useful upperbound is as follows:

$$x \geq 1 \Rightarrow \ln x \leq \frac{x-1}{\sqrt{x}} \tag{5}$$

So as long as $v_0 \geq v_{\text{stop}}$ (otherwise the problem is trivial) we can use this to upperbound the distance we will travel, and therefore lowerbound the value of x_{dock} as

$$x_{\text{dock}} \geq \frac{\frac{v_0}{v_{\text{stop}}} - 1}{C_d A \sqrt{\frac{v_0}{v_{\text{stop}}}}}$$

To begin proving this, we first show that (1) is a differential invariant of our system us the differential ghost method described in section 4.3.1, Proving Asymptotic Behavior. In this case, we use

$$\begin{aligned} b &= 1 \\ b' &= C_d A \left(v + \frac{v_0}{C_d A v_0 t + 1} \right) b \end{aligned}$$

To prove that

$$\begin{aligned} \exists b.\ b > 0 \wedge b \left(v - \frac{v_0}{C_d A v_0 t + 1} \right) = 0 &\Rightarrow v - \frac{v_0}{C_d A v_0 t + 1} = 0 \\ &\Rightarrow v = \frac{v_0}{C_d A v_0 t + 1} \end{aligned}$$

There is a small caveat. We *also* need to prove that $b > 0$ is always true. Unfortunately, we can't use straightforward differential induction for this either since $b' > 0$ is only true if $b > 0$ which is the thing we're trying to prove. So we need yet another differential ghost:

$$\begin{aligned} c &= \frac{1}{b} \\ c' &= -\frac{1}{2} C_d A (v + \frac{v_0}{C_d A v_0 t + 1}) \end{aligned}$$

And show that the following is also a differential invariant:

$$bc^2 = 1$$

Once we've proved all that, we can use the fact that $v \geq v_{\text{stop}}$ and (3) to include the invariant

$$t \leq \frac{v_0 - v_{\text{stop}}}{C_d A v_0 v_{\text{stop}}} \tag{6}$$

Finally, we add the following invariant that follows from (4) and (5)

$$x \leq \frac{v_0 t}{\sqrt{C_d A v_0 t + 1}}$$

which finally allows us to use (6) and conclude that

$$x \leq x_{\text{dock}}$$

is true at least until $v < v_{\text{stop}}$.

5.4 Efficiency

A common question to ask in response to a proof that something is safe is whether or not the safe controller is also efficient. As I showed in Section 5.2, there is an exact solution to this problem, so we can compare the efficiency of our controller to the efficiency of an optimal controller. Figure 4 shows the required docking distance for the safe controller versus an optimal controller at various starting speeds. The data was generated assuming $C_d A = 1$. In Figure 4 we see that for these relatively low speeds (keep in mind, this is just the speed when docking, not when we're driving in open water), our safe controller is between 40-70% as efficient as an optimal controller.

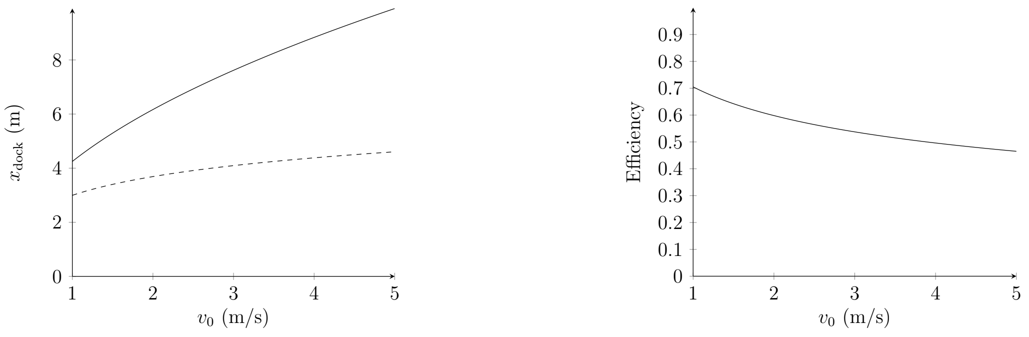

Figure 4: Required docking distance for the safe controller (solid) vs.optimal controller (dashed) (left). Example of manuevering between regions in a complex environment (right)

6 Conclusion

The main contributions of this project are two fold. One, I present a simplified model of surface vehicles that lends itself well to proofs and formal models. Second, using this model, I provide three models that were proved:

- Safe driving in rectangular regions
- Driving within complex regions
- Safe docking in one dimension

Each model is provided as a `.kya` file.

Future Work In the future, this work could be extended to include docking in two dimensions or fuel safety as was originally intended. Aditionally, the model itself could be improved to include more realistic dynamics such as drift. Also interesting would be to consider the effects of weather such as waves or wind and to take a hybrid games approach to prove that a controller can be safe even with such adverse conditions.

Evaluation The goal of this project was to design controllers that prove safety of various tasks commonly performed by boats. In particular, my original proposal outlined the following goals:

- Driving in a rectangular regions while maintaining a safe acceleration
- Avoiding static obstacles within the safe region
- Achieving docking in one dimension
- Achieving docking in two dimensions
- Maintaining fuel safety, and returning to a dock when the boat would otherwise run out of fuel

In the end, I managed to prove only the first of these three goals. The biggest lesson learned is that it is very hard to reason about a system without exact solutions (or, without exact solutions that KeyMearaX can reason about).

References

[1] Florida-Caribbean Cruise Association American Association of Port Authorities, 2016.

[2] J. Evans, P. Redmond, C. Plakas, K. Hamilton, and D. Lane. Autonomous docking for intervention-auvs using sonar and video-based real-time 3d pose estimation. In *Oceans 2003. Celebrating the Past ... Teaming Toward the Future (IEEE Cat. No.03CH37492)*, volume 4, pages 2201–2210 Vol.4, Sept 2003.

[3] Rose George. *Ninety Percent of Everything: Inside Shipping, the Invisible Industry That Puts Clothes on Your Back, Gas in Your Car, and Food on Your Plate.* Metropolitan Books, 2013.

[4] U.S. Navy. Status of the navy. Technical report, U.S. Navy, 2017.

[5] Robert Sutton Wasif Naeem and John Chudley. Modelling and control of an unmanned surface vehicle for environment monitoring. 2006.