Reachability Analysis for Robotic Motion Planning using Techniques from Verification of Hybrid Systems

Salvatore Candido
Department of Electrical and Computer Engineering, Beckman Institute
University of Illinois at Urbana-Champaign, USA
candido@illinois.edu

Abstract—We consider verification of the property that a control policy, with a specific robot system in a known environment, will lead to collision-free paths that always reach a goal. This is a stepping stone towards combining heuristic policies with verification to avoid computing optimal state feedback policies. We first discuss the difficulty of the Navigation Problem and why we turn to state feedback policies. We then analyze two robot systems: a robot whose continuous state is in \( \mathbb{R}^2 \) with rectangular constraints and a mobile robot whose continuous state is in \( SE(2) \) whose policy is a series of waypoints. Finally, we discuss application to more complex systems and high level goals of this line of research.

I. INTRODUCTION

The Navigation Problem [1] is central to the study of robotics, and the design of robotic systems. It is the task of finding a sequence of control inputs to move a system from a given start state to a state belonging to the set of goal states. Unlike the typical control problem, we also consider non-differential constraints on the state space in the form of forbidden, or obstacle, regions. A policy is considered to have failed if the system’s state enters one of these obstacle regions at any point during the state trajectory. Even computing policies for simplified versions of this problem are very difficult. The Path Planning Problem has the same task as the Navigation Problem, but considers the restriction of a holonomic system; a system without differential constraints. If our state space is denoted \( X \), the task is to find a function \( c : [0,1] \rightarrow X \) where \( c(0) \) is the starting state and \( c(1) \) is in the set of goal states. The Path Planning Problem has been shown to be PSPACE-Complete [2] and the best known algorithm has a time complexity of

\[
p^m \cdot \log p \cdot w^{O(m^2)}
\]

where \( m \) is the dimension of the state vector (degrees of freedom of the robot) and \( p \) is the number of algebraic constraint equations needed to represent obstacle regions in the state space. Clearly, even in this drastically simplified context, this a computationally expensive process.

An additional source of difficulty is uncertainty in the process and observation models. In the Path Planning Problem, we consider a deterministic, holonomic system; real robotic systems tend to be neither. One method to deal with uncertainty is to use a state feedback policy. A state feedback policy \( \Phi \) maps every \( x \in X \) to a control action. Essentially, a state feedback policy encodes a trajectory from every state to a state in the set of goal states. Thus, if the robot’s state deviates from the nominal trajectory, we can take this information into account and possibly modify future control inputs. Value and Policy Iteration are two widely used methods to construct these plans.

Unfortunately, we cannot use state feedback directly, as uncertainty in the system’s observation model means that, in general, the system does not have perfect knowledge of the state when computing the control action from the policy. In fact, in most cases we will have unobservable modes in our state vector and the ones that are observable will be uncertain.

Consider the problem of choosing a control input \( u \) to stabilize to zero a scalar system that evolves according to the differential inclusion

\[
\dot{x}(t) \in [u(t) - \epsilon, u(t) + \epsilon]
\]

for some \( \epsilon > 0 \). State feedback is one method to achieve this goal. If we can observe the state perfectly (our observation \( y(t) = x(t) \) for all \( t \geq 0 \), we can choose \( u(t) = -y(t) = -x(t) \) and it can be shown that \( x(t) \in [-\epsilon, \epsilon] \) for all \( t \geq 0 \). If \( \epsilon \) is small, this is typically acceptable performance.

In contrast, consider the case of an unobservable state. Since we have no idea of the state trajectory, a \( u(t) \neq 0 \) could either drive \( x(t) \) closer or farther away from zero. If we have no a priori knowledge of which trajectories are more likely, our control input is useless and if \( u(t) = 0 \) for all \( t \geq 0 \) then the possible state trajectories are \( x(t) \in [-\epsilon t, +\epsilon t] \). As \( t \rightarrow \infty \), we become increasingly uncertain of \( x(t) \).

In practice, we do not have access to perfect state information, as in the first situation. Sensors typically have some associated noise, i.e. \( y(t) \in [x(t) - \gamma, x(t) + \gamma] \), and we can only retrieve observations at discrete times rather than over the continuum, i.e. \( y(t) = x(t) \). In this example, consider the case where \( y(t) \in [x(t) - \gamma, x(t) + \gamma] \). We can show that
using the policy \( u(t) = -y(t) \) that
\[
\dot{x}(t) \in [u(t) - \epsilon - \gamma, u(t) + \epsilon + \gamma]
\]
so we can show that \( x(t) \in [-\epsilon - \gamma, \epsilon + \gamma] \) for all \( t \geq 0 \).

Our ultimate aim is to avoid some of the computation by modifying an existing state feedback policy or using another heuristic method to compute a policy. A large number of methods have been developed for use on specific types of systems or under specific conditions. Heuristic methods can be significantly more efficient, but do not guarantee desirable results. This generate then check scheme will consequently require a method to verify that our policy achieves acceptable performance. Specifically, we will then need to verify that our system reaches and remains in a set in the goal set and guarantee obstacle avoidance along the way for a set of trajectories. Furthermore, while state feedback compensated for the problem of uncertainty in the process model, it increases the complexity of the planning problem because it requires we compute trajectories that pass through every point in the state space. Computing a state feedback policy over the entire state space is always expensive, and in many cases infeasible. Complicating the problem further, every time the set of goal states changes, a new policy is needed.

In this paper, we discuss the analysis of two robot systems. The first is a robot whose configuration is in \( \mathbb{R}^2 \) and can only move in four cardinal directions. The obstacle constraints, the regions of the policy, and the observation uncertainties are rectangular. We then consider a robot whose configuration is \( SE(2) = \mathbb{R}^2 \times S^1 \). This system is nonlinear, the observation model is no longer rectangular, and the entire state vector is no longer observable. Finally, we discuss our results and some future directions for this work.

II. PROBLEM STATEMENT

Given a system with
- continuous state \( x \in X \), the configuration and possibly velocity of the robot
- an observation model \( y = h(x) \)
- a function (possibly a recursive filter) for estimating state \( \hat{x} \in X \) based on process and observation models
- a policy \( \Phi \) which maps estimated state to the appropriate controller
- an environment \( \mathcal{W} \subset \mathbb{R}^3 \)
- a set of starting states \( X_{start} \)
- a set of goal states \( X_{goal} \)

we wish to analyze whether the robot’s continuous state will remain in \( \mathcal{W} \) and whether \( x(t) \in X_{goal} \) as \( t \to \infty \).

Define the predicates Collision(\( x \)) and In(\( x, R \)) where \( R \subset \mathbb{R}^2 \). The predicate In(\( x, R \)) is true only if \( x \in R \), and Collision(\( x \)) is true only if the state \( x \) induces a collision,
i.e., $\neg \ln(x, W)$. The main properties which we would like to verify are, for some starting set $X_0$

**P1.** $A \square \{\neg \text{Collision}(x)\}$

**P2.** $A \diamond \{\ln(x, X_{\text{goal}})\}$

**P3.** $A \square \{\ln(x, X_{\text{goal}}) \rightarrow x\ln(x, X_{\text{goal}})\}$

This says that all trajectories will be collision free, reach the goal set, and stay in the goal set. If we cannot guarantee these strong properties, we may also be interested in verifying that there exists a trajectory that is collision free and reaches the goal, although not necessarily every trajectory has this guarantee. We may also not require P3; reaching the goal once may be enough. Finally, we also may be interested in the inverse problem: Given a policy and the above state, from which states can one start such that P1, P2, and P3 hold.

## III. Mobile Robot with State in $\mathbb{R}^2$

As our first attempt, we explored the reachability properties of the simplest geometric path planning problem, a kinematic point robot moving on a plane. Consider the automaton shown in Figure 2. The robot’s state is $x \in \mathbb{R}^2$ and the free workspace of the robot is $W \subset \mathbb{R}^2$. The robot will start at a deterministic location $x_0$ and attempt to reach a state in the set $X_{\text{goal}} \subset W$. We model the uncertainty in the process model of the system as a differential inclusion, $\dot{x} \in F(x, u)$ where $F(x, u) \subset \mathbb{R}^2$. The estimate of the state $\hat{x}$ has the same deterministic dynamics as $x$ but updates every time a new sensor reading is taken. Every second, our system receives an observation $y$. Due to sensor uncertainty, the measurement is not exact but will fall within the region $y_i \in [x_i - \gamma, x_i + \gamma]$ for a constant $\gamma > 0$.

The system has five main controllers, move up, move down, move left, move right, and stop. The process uncertainty model is rectangular. Essentially, the robot has an uncertainty of $\pm \beta$ in its velocity along the direction of travel and $\pm \alpha$ in the orthogonal direction. These control actions define the coordinate axes of workspace, and the obstacle constraints are assumed to be rectangular with respect to those coordinates. If the obstacles are not strictly rectangular, a rectangular overapproximation can be used. Depending on the amount of overapproximation required, this may or may not be an appropriate model.

A policy $\Phi(\hat{x})$ maps the best estimate of the state to the control action that will be taken. We will represent the policy as a set of regions producing the same control action. Define an $R_\theta \subset \mathbb{R}^2$ as the maximal set including $x$ such that for all $x \in R_\theta$, $\Phi(x) = u$, and for every neighboring $x$ to $R_\theta$, $\Phi(x) \neq u$. For simplicity, we assume that every $R_\theta$ can be represented as a set of rectangular constraints. This assumption can be guaranteed in the construction of $\Phi$ and, in most cases, is not too limiting a constraint on the allowable set of policies.

Switching between the modes of the automata occurs in two cases. First, every time unit the robot takes a new sensor measurement and updates its estimate of its position $\hat{x}$ accordingly. It will set $\hat{x} = y$ assuming that $y$ is a reachable state for $x$. If it is not, $\hat{x}$ will be chosen as the point with the closest Euclidean distance to $y$ inside some approximation of the reachable set of $x$, $\text{Reach}(x)$. If the robot’s state estimate transitions from $R_\theta$ to $R'_\theta$, then again the robot transitions states. Since every $R_\theta$ can be represented as a set of rectangular constraints, we can represent the robot’s state estimate transitioning as a disjunction of these constraints. The predicate $B(x)$ is then a disjunction of conjunctions which is true if and only if the state estimate is on the boundary between regions.

This system falls into the class of rectangular initialized automata. Although, in general, the reachability of this class of systems is undecidable, we were able to find some conditions under which we guarantee reachability of the goal set. We model the hybrid system in a behavior-based manner. Essentially, identical behaviors are grouped into discrete states and the information about where each behavior is appropriate is stored on the transitions of the system. However, another construction, which is useful for analysis, is a location-based partition where the discrete states are the regions defined by $\Phi$.

Although we analyze a point robot, this model generalizes to some real-world systems. This model corresponds well with a cylindrical research mobile robot such as a iRobot Magellan or a Nomad SuperScout operating in a building. The obstacle constraints of this system can converted to the case of a point robot (and remain rectangular) by computing the Minkowski Sum [1] of the robot’s shape with the obstacles in the environment. This model also corresponds to a metal particle in a fluid being manipulated by a magnetic field. Both systems are available on the University of Illinois campus.

### A. Analysis

In this analysis we considered the case where $2\gamma > 1 - \beta$ and $1 - \beta > \alpha$. We will demonstrate a policy construction that satisfies the properties of interest in Section II; namely obstacle avoidance and guaranteed reaching of the goal.

Define the **nominal trajectory** to be the trajectory realized by $\dot{x} = u$ and $\dot{y} = x$ at every $t \geq 0$. This would be the trajectory of the deterministic, perfectly observable system or the “desired” trajectory. Define a **1-successor region** of $R_\theta$, $R'_{\theta}$ to be a region where there exists an $x \in R_\theta$ and a trajectory $\tau$ where $x \rightarrow \tau x'$ for $x' \in R'_{\theta}$ and no $x''$ along the trajectory $\tau$ belongs to any other region. Define an **i-successor region** of $R_\theta$, as a region $R'_{\theta,i}$ where there exists a sequence of $i$ regions where $R'_{\theta,i}$ is the $i$th region and the $j$th region is a 1-successor of the $(j - 1)$th region for $0 < j \leq i$. Then, we will say $R'_{\theta,i}$ is a **successor of** $R_\theta$ if it is an i-successor of $R_\theta$ for any finite $i$. We will call $R'_{\theta,i}$ a **direct successor** of $R_\theta$ if there exists an $x \in R_\theta$ and a nominal trajectory $\tau$ where $x \rightarrow \tau x'$ for $x' \in R'_{\theta,i}$ and no $x''$ along the trajectory $\tau$ belongs to any other region. We say that two regions $R_\theta$ and $R'_{\theta,i}$ are adjacent if for any $x \in R_\theta$, there exists a $\hat{x} \in R'_{\theta,i}$.

We will also define the predicates $\text{Successor}(R_\theta, R'_{\theta,i})$, $\text{Successor}(R_\theta, R'_\theta)$, and $\text{DirectSuccessor}(R_\theta, R'_{\theta,i})$. The predicate $\text{Successor}(R_\theta, R'_{\theta,i})$ is true if and only if $R'_{\theta,i}$ is an i-successor of $R_\theta$. Similarly, $\text{Successor}(R_\theta, R'_\theta)$ is true if $R'_\theta$ is an i-successor of $R_\theta$ for any $i$. The predicate
Fig. 2. Automaton for Robot with State in \( \mathbb{R}^2 \)

**Direct Successor** \( (R_{\Phi}, R'_{\Phi}) \) is true only if \( R'_{\Phi} \) is a direct successor of \( R_{\Phi} \).

Assume we have a policy which satisfies four construction rules, specified below. The claim is that under these conditions the system will always avoid obstacles and reach the goal. We first state the policy construction rules and then prove that the formulas hold under those rules.

**PCR1:** (Obstacle Safety Region) Compute the Minkowski sum of obstacle region \( i \) with a square of side length \( 1 + \beta + \gamma + \epsilon \) for any \( \epsilon > 0 \) and call that region, less the obstacle region, the safety region for obstacle \( i \). For every \( x \) in the safety region, define \( \Phi(x) \) so that the obstacle region is not a 1-successor of the region containing \( x \) and so that \( \Phi \) does not violate PCR3.

**PCR2:** (Invariant Goal Set) The goal region should have length and width at least \( 2\gamma + \epsilon \) for any \( \epsilon > 0 \). Remove a length of \( \gamma \) from each side and give it a policy pointing into the center region. The invariant center region should have a stop position.

**PCR3:** (Progress Guarantees) For every \( R_{\Phi} \), there should be no adjacent \( R'_{\Phi} \) with a policy whose nominal trajectory is opposite the nominal trajectory of \( R_{\Phi} \).

**PCR4:** (Policies Flow to Goal) The region \( X_{\text{goal}} \) should be reachable from every \( R_{\Phi} \) in \( W \) using the nominal trajectory. Furthermore, no \( R_{\Phi} \) should be an \( i \)-successor to itself for any \( i > 1 \) and no \( R_{\Phi} \) other than \( X_{\text{goal}} \) may use the stop controller.

PCR1 requires there to be a “buffer” around obstacles which the robot cannot move completely through without detecting its proximity to the obstacle. PCR2 gives the minimum size of the goal set if the robot is expected to be able to eventually detect it is in the goal set and stop under all conditions. PCR3 restricts the organization of \( \Phi \) in the sense that controllers that oppose one another should not be too close to another so a chattering condition does not occur. In also restricts the organization of \( \Phi \) at a higher level; justifying the expectation that we should have large groups of states neighboring one another whose associated control action is the same. PCR4 tells us there should be no cycles in the directed graph defined over neighboring policy regions. If there are, there can be period orbits or infinite non-periodic trajectories in the state trajectory that do not reach the goal set. These conditions are difficult to find using software tools or even analysis by hand.

The structure of these construction rules are not unexpected. However, each rule gives specific conditions in terms of the system parameters under which collision avoidance and guaranteed reachability of the goal will be satisfied. If the rules PCR1, PCR2, and PCR3 are not satisfied, a specific counterexample can given to show that our properties of interest cannot be satisfied. However, the rules remain sufficient and not necessary because this example will require certain starting conditions, i.e. the system may never reach the point of concern from a given starting state. For this reason it may be useful to employ model checking software to check trajectories emanating from specific initial states for cases where these policy rules generally hold, but there are a few exceptions.

**Lemma 1.** \( A [\neg \text{Collision}(x)] \) if policy construction rule PCR1 is used.

**Proof:** Consider Fig. 3(a), a depiction of the safety regions after PCR1 is applied. We will use the Inductive Invariance technique [3] to show that \( \text{Collision}(x) \) will never be true starting from any \( x_0 \in W \). First, the continuous state variables \( (x) \) are continuous in time and they do not reset under any actions. Thus, under any action \( a, x \rightarrow^a x \) so \( \neg \text{Collision}(x) \) is invariant under actions.

Consider a state \( q \) with \( q.x \) within the inner safety region (points whose \( L_\infty \) distance from the obstacle boundary are
greater than or equal to $1 + \beta + \epsilon$ and $c = 0$, the corresponding policy specified by $\Phi(\hat{x})$ for any possible $\hat{x}$ will have a nominal trajectory pointing away from the obstacle boundary. Thus, under this trajectory the collision free property is preserved.

Consider $q'$ the case when $q', x$ is outside the inner safety region. Under the worst case, let $\hat{x}$ correspond to a position outside of the safety region and $\Phi(\hat{x})$ specify a nominal trajectory moving directly towards the obstacle boundary. The trajectory can evolve for a maximum time of one time unit as at $c = 1$ a new sensor reading will occur. Under this trajectory, the robot can move at most $1 + \beta$ towards the obstacle boundary. Since $q' \rightarrow^\tau \tau''$ begins in the outer safety region it is at least $1 + \beta + \epsilon$ from the obstacle boundary along this direction of travel so the collision free property is preserved under trajectories from these states.

Consider $q_0$, where $q_0, x$ is outside the safety region. Using a similar argument to the above paragraph, we see that under no trajectory can we reach the obstacle without encountering a $q$ that is considered two paragraphs above.

This implies that the condition free property is preserved under all actions and trajectories starting from all states in the policy. Thus, the robot will never be in collision with an obstacle.

Although we assumed a region with arbitrary policy can border the safety region, we know that certain policies will violate PCR3. If we assume that the policy additionally meets PCR3, we can reduce the length of the safety region to $\alpha + \epsilon + \gamma$ which may be a considerable reduction.

In the next proof, we will construct a policy graph $G_\Phi$ where each $R_{\Phi}$ defined by $\Phi(\cdot)$ is a vertex and the 1-successor relationships between regions are the directed edges of the graph. We will use this construction to analyze the behavior of the system.

**Lemma 2.** $A \diamond \{\text{In}(x, Q_{\text{goal}})\}$ if policy construction rules PCR3 and PCR4 are met.

**Proof:** From PCR4, $G_\Phi$ is structured so that every $R_{\Phi}$ has a path to $X_{\text{goal}}$, there are no cycles, and the system only stops upon reaching $X_{\text{goal}}$. Thus, it is sufficient to show that at least one graph transition will occur for $q$ with $x \notin X_{\text{goal}}$. We demonstrate this by showing that the robot cannot persist indefinitely in any non-goal region.

For any two adjacent regions under PCR3, the directions of the nominal trajectories can only be orthogonal to one another or the same direction. Consider the case where both nominal trajectories are pointing the same direction, and without loss of generality let it be the controller corresponding to $\dot{x}_1 \in [1 - \beta, 1 + \beta], \dot{x}_2 \in [-\alpha, +\alpha]$. Then, the rightmost region $R^2_{\Phi}$ is a 1-successor the region to the left $R^1_{\Phi}$. For any $x$ in $R^1_{\Phi}$ sufficiently far from other regions, $\Phi(x, \dot{x})$ will be move right. Thus, the robot will progress at least $1 - \beta$ towards $R^2_{\Phi}$. After a sufficient number of steps, the distance to $R^2_{\Phi}$ will shrink to zero and we must reach a $q''$ where $q'', x \in R^2_{\Phi}$.

Consider Figure 4, two adjacent regions who are 1-successors of each other but their nominal trajectories are orthogonal to one another. Without loss of generality, consider the policy in the leftmost, starting region $R^1_{\Phi} = R^{11}_{\Phi} \cup R^{12}_{\Phi}$ to be $\dot{x}_1 \in [1 - \beta, 1 + \beta], \dot{x}_2 \in [-\alpha, +\alpha]$ and the policy in the rightmost, transitioned-to region $R^3_{\Phi} = R^{21}_{\Phi} \cup R^{22}_{\Phi}$ to be $\dot{x}_1 \in [-\alpha, +\alpha], \dot{x}_2 \in [1 - \beta, 1 + \beta]$, i.e. move right, move up. If the robot is within $\gamma$ of the border of the two regions, because of sensor uncertainty moving either right or up is possible. Otherwise, the control is fixed. If the robot reaches an $x \in R^{22}_{\Phi}$, then the robot will never return to the $R^1_{\Phi}$. If the robot has $x \in R^{11}_{\Phi}$, then the robot must either visit the boundary region or exit the $R^3_{\Phi}$ to a different 1-successor, as the robot will always make progress of $1 - \beta$ towards $R^3_{\Phi}$ and possibly progress orthogonal to that direction.

Consider the function $V = d_1(q) + d_2(q)$ where $d_1(q)$ is the shortest distance from $q, x$ to any point in $R^{11}_{\Phi} \cup R^{12}_{\Phi}$ and $d_2(q)$ is the shortest distance from $q, x$ to any point in an additional adjacent region above the pictured regions. If the robot has $q$ such that $q, x \in R^{12}_{\Phi} \cup R^{21}_{\Phi}, \Phi(q, \dot{x})$ will either have a nominal trajectory of up or right. If it is to move right, $\dot{x}_1 \geq 1 - \beta$ and...
\[ \dot{x}_2 \geq -\alpha. \] Since \( 1 - \beta > \alpha \), \( \dot{V} < 0 \). By the same argument for the move up controller, we see that \( \dot{V} < 0 \), as well. Thus, since \( V \) is always decreasing under any \( q \rightarrow q' \) for any \( q \) such that \( q, x \in R^2_0 \cup R^2_k \) and \( V \) does not increase under actions, we know that \( V = 0 \), eventually. Since the \( V \leq 1 - \beta - \alpha < 0 \), we know \( V = 0 \) occurs in a bounded amount of time. Finally, \( V = 0 \) if and only if the robot has reached either \( R^2_0 \) or another 1-successor. Thus, the robot cannot persist in a continuous portion of the state. Thus, if the property held a any action \( \Phi(\hat{x}) \rightarrow x \Phi(x, X_{goal}) \) is met.

\[ x \] is always decreasing under any \( \tau \) with a policy whose nominal trajectory is opposite the nominal trajectory of \( R \), a trajectory which never reaches the goal exists. This will be due to a chattering phenomenon with the robot’s state estimate switching between states where each state uses the policy opposite to the one in the previous state. Because of this, the robot may never leave the set of states where state estimates with both policies are possible. If there is an \( R \) that is an \( i \)-successor to itself where \( i > 1 \), there is possibly a trajectory which never reaches the goal but it is not guaranteed.

**Lemma 3.** \( \Box \{ I(x, X_{goal}) \rightarrow xI(x, X_{goal}) \} \) if policy construction rules PCR2 and PCR4 is met.

**Proof:** Consider Fig. 3(b), a depiction of the goal region after PCR2 is applied. We will use the Inductive Invariance technique to show the lemma.

First, the continuous state variables \( x \) are continuous in time and they do not reset under any actions. Thus, under any action \( a \), \( x \rightarrow a x \) and the predicate depends only on the continuous portion of the state. Thus, if the property held prior to any action, it holds after that action. Under PCR4, every neighboring region to \( X_{goal} \) has a nominal trajectory pointing in to \( X_{goal} \). Thus, for any \( x \in X_{goal} \) within \( \gamma \) of another region, any possible \( \Phi(\hat{x}) \) will result in \( x \rightarrow \tau x' \) with \( x' \in X_{goal} \).

If \( x \in X_{goal} \) and \( x \) is more than \( \gamma \) away from any other region, then under any possible \( \hat{x} \) the policy will be \( \hat{x} = 0 \). This means that under all trajectories, \( x(t) = x \) for \( t \geq t_i \). Thus, if \( x(t_i) \in X_{goal} \) for any \( t_i \), then \( x(t) \in X_{goal} \) for all \( t \geq t_i \).

We have shown that P1, P2, and P3 hold individually under some subset of PCR1, PCR2, PCR3, and PCR4. We now state this explicitly in a proposition.

**Prop 1.** The properties P1, P2, and P3 are satisfied for any starting state in \( W \) of hybrid systems of the form discussed in Section III with \( 2\gamma > 1 - \beta \) and \( 1 - \beta > \alpha \) if \( \Phi \) follows construction rules PCR1, PCR2, PCR3, and PCR4.

**Proof:** This follows directly from Lemmas 1, 2, and 3.

**B. Discussion**

The result of this analysis was intuitive and no surprising results were shown. However, analysis of this system gave an good example of verification techniques for a robotic motion planning system. Additionally, the analysis yielded quantitative relationships between policy, sensor quality, and process (actuator) quality. Furthermore, it showed that we can verify reachability properties by checking only the policy and basic properties of the system. This is a much less computationally intensive task then the general reachability problem for this class of systems, which may even be undecidable.

One point of concern in the analysis is the restriction to the case where \( 2\gamma > 1 - \beta \) and \( 1 - \beta > \alpha \). If the first condition does not hold, the uncertainty in the sensing model is less than the distance traveled between samples. This means the sensor is very reliable and there will be very few areas where there is an ambiguity in the policy. Analysis will be similar to that for a deterministic sensing model which is uninteresting for this analysis. Thus, an analysis of this sort does not make sense for the class of systems where this property does not hold. This second condition requires that we will always make more progress in the nominal or desired direction than progress due to noise. Without this condition, we can almost never guarantee success under any policy and this condition will typically be satisfied in experimental and real-world conditions on robotics hardware.

A relationship was shown between the size of the goal set and the quality of the sensor. If side length of the goal set is not as least \( 2\gamma + \epsilon \), the robot will not be guaranteed to detect that it is within the goal and thus may never stop. Another relationship was between sensor quality and proximity to obstacles. We must have some minimum safety region near obstacles or the robot, for some trajectories, will be able to hit the obstacles. The gives us a hard limit on narrow passages through which the robot can move. Finally, this analysis explicitly defined the structure of policies if we want to guarantee the robot reaches some goal set. Any opportunity for the robot to be caught in a cycle or stop at a state not in goal will prevent guarantees of reaching the goal set.

While the conditions shown above are more restrictive than those typically seen in robotic planning algorithms, it is important to note that this is a worst case analysis. Often, stalling or periodic trajectories outside the goal set correspond to what would typically be considered low probability events in typical stochastic models of sensor and process noise. Also, this analysis holds for all \( x \in W \). If we were willing to limit the possible starting positions of the robot, we could possibly eliminate a subset of \( W \) that can never be reached. In this case, the policy at those locations would be irrelevant. Thus, by combining this analysis with forward simulation techniques, we may be able to additionally verify a larger class of policies that do not necessarily meet all the rules at every location in the policy.

These properties were shown using inductive invariance methods. While useful in this case, it become significantly more complicated to prove P1, P2, and P3 for more complex models using these properties.
IV. PLANAR MOBILE ROBOT WITH ORIENTATION

A more general model of a mobile robot is one with continuous state modeled as a point in $SE(2) = \mathbb{R}^2 \times S^1$; a robot moving in a planar environment with orientation. This would correspond to systems where the robot is of a non-circular shape or where the robot can only translate in a certain direction; like most mobile robots. We will denote the robot’s state as $x = (x_1, x_2, \theta)$.

Consider Figure 5, a diagram describing the automaton representing this robot system. The robot has two controllers. One controller allows the system to move forward deterministically with a fixed orientation and is described by

$$\dot{x} = [\cos \theta, \sin \theta]^T$$
$$\dot{\theta} = 0$$

(4)

(5)

The other is a pure rotation with uncertainty in the process model.

$$\dot{x} = 0$$
$$\dot{\theta} \in [u - \epsilon, u + \epsilon]$$

(6)

(7)

where $u = \{+1, -1\}$ corresponding to turning left and right. This is a good approximation to a mobile robot with fairly good wheel encoders being used in the low-level controllers.

We will first discuss the system in open loop, ignoring the sensor. This will illustrate the difficulty of computing the reachable set of this system after a sequence of switching events. In the following section, we will make use of the position sensor. In this case, every $t_y$ time units using the move forward controller, the sensor takes a new measurement. Since the observation does not measure the rotational component of the system it does not help to take measurements while rotating. An observation $y \in \mathbb{R}^2$ from the sensor is guaranteed to be within $\delta$ of the actual position of the robot. Thus, $y \in \{ z \in \mathbb{R}^2 : ||z - x|| < \delta \}$.

After some time has elapsed, we will track the reachable set of states of the automaton $Reach_A(t)$. We will discuss computing this set explicitly and then an over-approximation. Our state estimate $\hat{x} = (\hat{x}_1, \hat{x}_2, \hat{\theta}) \in Reach_A(t)$ evolves deterministically during between sensor measurements. When using the move forward controller, $\hat{x}_1 = \cos \hat{\theta}$ and $\hat{x}_2 = \sin \hat{\theta}$.

When using the rotation controller, $\dot{\theta} = u$. After a sensor action, if the sensor reading $y \in \mathbb{R}^2$ is in $Reach_A(t)$ (or our over-approximation) projected onto $\mathbb{R}^2$ then we will reset the Euclidean portion of our state estimate to $(\hat{x}_1, \hat{x}_2) = (y_1, y_2)$. If not, we will choose $(\hat{x}_1, \hat{x}_2)$ as the point with smallest $L_2$ distance to $y$ in our reachable set.

Our policy in this case will be state feedback with respect to a set of waypoints. We will mark a sequence of waypoints $p_1, p_2, \cdots, p_\text{n}$ where each $p_i \in \mathbb{R}^2$ in the workspace. The robot will attempt to move from one waypoint to the next. The waypoint $p_i$ will be placed in $X_{goal}$ and the robot will “reach” waypoint $p_i$ and attempt to move towards $p_{i+1}$ when the robot’s state estimate is within $\delta$ or $p_i$. The general idea of the state feedback policy will be to rotate so that the robot’s forward controller moves along the line from $\hat{x}$ to $p_i$ and then use the move forward controller until $\hat{x}$ reaches a state within $\delta$ of $p_i$. We will use this policy exactly when evaluating reachability under open loop. Then we will modify it slightly when we begin using the sensor.

We will consider the open loop (no sensor) case briefly in order to demonstrate the difficulty of computing the reachable set of this nonlinear system. We will then move on to a more useful analysis which over-approximates the reachable set and bounds the trajectories of the robot within tubes defined by the waypoints. Obstacle avoidance can be guaranteed by placing the waypoints so that the tubes do not collide with any obstacles. With this simple check for obstacle avoidance, we can heuristically generate a waypoint plan and quickly check to see if it will suffice.

A. Open Loop Reachability

This system has nonlinear dynamics and as a result, computing the reachability is significantly more difficult than
the previous case where \( x \in \mathbb{R}^2 \) and the dynamics were rectangular sets. First consider open-loop waypoint control, i.e. no sensors and a preplanned alternation of moves forward for a length of times \( d_1, d_2, \ldots, d_n \) where \( d_i \in \mathbb{R} \) and rotations \( r_1, r_2, \ldots, r_n \) where \( r_i \in [-\pi, \pi] \). Since we have no sensors, there is no way to curtail the growing uncertainty pushed into \( r \) without sensing, an upper bound will typically be obvious and remains open.

One technique to avoid this exponential growth is to upper bound the reachable set with a set with simpler representation. However, without sensing, an upper bound will typically be too loose to be useful in the presence of obstacles. We then turn to an analysis of the system with a position sensor where a useful upper bound can be established. However, this example shows the difficulty of reachable sets of even simple hybrid nonlinear systems.

**B. Waypoint Analysis with Position Sensor**

Consider Figure 5, our automaton will be as discussed earlier with a new closed-loop policy. One thing to note is that we only get new observations every \( t_y \) time units of using the move forward controller. Since our observation only gives us information about position, new observations after just rotation are not guaranteed to give us new information.

![Fig. 6. Open Loop Reachability of the System](image)

Let \( \Delta \theta \) be the amount of uncertainty in orientation, i.e. \( \theta \in [\theta_{\text{min}}, \theta_{\text{max}}] \Rightarrow \Delta \theta = \theta_{\text{max}} - \theta_{\text{min}} \). Similarly, let \( \Delta x_1 = x_{1,\text{max}} - x_{1,\text{min}} \) and \( \Delta x_2 = x_{2,\text{max}} - x_{2,\text{min}} \). Thus, we have three sets \( X_1, X_2, \text{and } \Theta \) defined by maximum and minimum possible coordinates and our over-approximation of the reachable set is \( X_1 \times X_2 \times \Theta \). Let \( \theta_d \) be the angle between the vector defined by the robot’s estimated position and center of the robot’s uncertainty in orientation, (\( \theta_{\text{max}} - \theta_{\text{min}} \))/2 and the \( x_1 \) axis.

The policy component of the automaton is given by Algorithm 1. Essentially, the robot estimates its position and computes it over-approximation of uncertainty based on new observation. It then adjusts its orientation so that the center of its uncertainty in \( \theta \) is aligned with the ray pointing from the robot’s estimate of its position to the waypoint. It then moves forward until it receives a new observation and repeats until it meets the criterion to have “reached” the waypoint.

We will analyze this system by first examining the movements of the system between a starting state and next waypoint. By showing we are guaranteed to reach the next waypoint under certain conditions, we can then draw conclusions iteratively about a sequence of waypoints. Consider Figure 7. Without loss of generality, we will define the starting state \( q_0, x \) of the robot to be \((0, 0)\) and let the waypoint be \((0, x^d)\). We know the actual starting position \( q_0, x \) will be within \( \delta \) of \((0, 0)\). The \( x_1 \) coordinate will correspond to the distance towards the waypoint traveled, and \( x_2 \) to the distance away from the \( x_1 \) axis. Also, we will assume that the set of uncertainty in \( \theta \) is centered along the \( x_1 \) axis.

We now prove that we can bound the reachable set of projection of the robot onto \( \mathbb{R}^2 \) in sets around the waypoints and line segments connecting the waypoints. An overview of the proof is to show

1. \( \Delta x_1, \Delta x_2 \) are bounded for all \( n \geq 0 \).
2. \( \Delta \theta(t) \leq \Delta \theta(0) < \pi \) for all \( t \geq 0 \) under the specified policy.
3. \( \theta_d \pm \frac{\Delta \theta}{2} \) is bounded away from \( \pm \frac{\pi}{2} \).
4. For sufficiently large \( x_1, \Delta \theta/2(x_1) \leq \tan \left( \frac{2\delta}{x_1} \right) \).
5. \( x_{2,\text{max}}, x_{2,\text{min}} \) are bounded with respect to \( x^d, x_1 \), and \( \delta \).

Item (3) allows us to guarantees progress towards the waypoint at every step. Item (4) shows there will be a reduction in the uncertainty if there is enough distance between waypoints. This is somewhat analogous to a dwell time condition. Finally,

**Algorithm 1: Policy for Waypoint System**

Compute \( X_1(t), X_2(t), \text{and } \Theta(t) \);

if \([x_1 - x^d] < \delta \):
  - Begin Next Waypoint;
else if \( \Delta \theta(t) < \Delta \theta(t - t_y) \):
  - Rotate the system so that \( \theta_d \) points towards \((x^d, 0)\);
  - Move forward for \( t_y \) seconds;
else
  - Move forward for \( t_y \) seconds.

---

\[ \text{Reach}_{\Delta}(t = d_1) \]
\[ \text{Reach}_{\Delta}(t = d_2) \]
item (5) allows us to bound the reachable sets in \( \mathbb{R}^2 \). It is important to do so because we can then collision check those geometric objects to ensure they do not intersect an obstacle in our workspace map. If no collisions occur, we are guaranteed to have a collision-free trajectory that reaches the goal.

**Lemma 4.** \( \Delta x_1, \Delta x_2 \) are bounded for all now \( t \geq 0 \).

**Proof:** We will show this property using the Inductive Invariance technique. In \( q_0 \), by assumption, \( x_{1, \max} = x_{2, \max} = -x_{1, \min} = -x_{2, \min} = \delta \) so \( \Delta x_1 = \Delta x_2 = 2\delta \).

When an observation is taken, we update \( x_{1, \max}, x_{2, \max}, x_{1, \min}, \) and \( x_{2, \min} \). This is done according to the equations

\[
\begin{align*}
x_{1, \max}(\text{now}) &= \min \{ x_{1, \max}(\text{now}^-), y_1 + \delta \} \quad (8) \\
x_{1, \min}(\text{now}) &= \max \{ x_{1, \min}(\text{now}^-), y_1 - \delta \} \quad (9) \\
x_{2, \max}(\text{now}) &= \min \{ x_{2, \max}(\text{now}^-), y_2 + \delta \} \quad (10) \\
x_{2, \min}(\text{now}) &= \max \{ x_{2, \min}(\text{now}^-), y_2 - \delta \} \quad (11)
\end{align*}
\]

No other actions modify these quantities so \( \Delta x_1 \) and \( \Delta x_2 \) are decreasing under actions, and thus bounded.

Under trajectories specified by the Turn mode \( x_1 = x_2 = 0 \) so \( \Delta x_1, \Delta x_2 \) do not change. Under trajectories specified by the Forward mode, for any \( t \in (kt_y, (k+1)t_y) \) for \( k \in \mathbb{Z}^+ \),

\[
\begin{align*}
x_{2, \max}(t) &= x_{2, \max}(kt_y) + (t - kt_y) \sin[\theta_{\max}(kt_y)] \quad (12) \\
x_{2, \min}(t) &= x_{2, \min}(kt_y) + (t - kt_y) \sin[\theta_{\min}(kt_y)] \quad (13)
\end{align*}
\]

Thus,

\[
\begin{align*}
\Delta x_2(t) &= x_{2, \max}(t) - x_{2, \min}(t) \\
&= \Delta x_2(kt_y) + (t - kt_y) \{ \sin[\theta_{\max}(kt_y)] - \sin[\theta_{\min}(kt_y)] \} \\
&= \Delta x_2(kt_y) + 2(t - kt_y) \cos[\theta_d(kt_y)] \sin[\Delta \theta(kt_y)]/2 \\
&\leq 2\delta + 2t_y
\end{align*}
\]

Thus \( \Delta x_1 \) and \( \Delta x_2 \) are bounded under trajectories.

**Lemma 5.** \( \Delta \theta(t) \leq \Delta \theta(0) < \pi \) for all \( t \geq 0 \) under the specified policy.

**Proof:** We will show this property using the Inductive Invariance technique. In \( q_0 \), by assumption, \( \theta_{\max}(0) < \frac{\pi}{2} \) and \( \theta_{\min}(0) > -\frac{\pi}{2} \) so \( \Delta \theta(0) < \pi \).

Under the trajectory described by the Forward mode, \( \dot{\theta} = 0 \) so \( \theta_{\min} \) and \( \theta_{\max} \) will not change. Thus, \( \Delta \theta(t) = \Delta \theta(kt_y) \) for any \( t \in (kt_y, (k+1)t_y), k \in \mathbb{Z}^+ \).

Under the specified policy, we only reach the Turn mode if and only if \( \theta_{\max} \) or \( \theta_{\min} \) change. We will show that \( \Delta \theta \) is reduced under any execution containing a trajectory with the Turn mode. When we take a sensor measurement, we set

\[
\begin{align*}
\theta_{\max}((k+1)t_y) &= \min \{ \theta_{\max}(kt_y), \sin\left(\frac{x_{2, \max}((k+1)t_y) - x_{2, \min}(kt_y)}{t_y}\right) \} \quad (18) \\
\theta_{\min}((k+1)t_y) &= \max \{ \theta_{\min}(kt_y), \sin\left(\frac{x_{2, \max}(kt_y) - x_{2, \min}((k+1)t_y)}{t_y}\right) \} \quad (19)
\end{align*}
\]

Thus, \( \theta_{\max} \) is decreasing and \( \theta_{\min} \) is increasing. This means \( \Delta \theta \) is decreasing. Let \( r \) be the net decrease, i.e. \( \Delta \theta((k+1)t_y) = r + \Delta \theta(kt_y) \) for some \( r \in [0, \Delta \theta(kt_y)] \). The maximum rotation in the Turn mode under our policy is \( r/2 \). (Consider the case when only one of \( \theta_{\min} \) or \( \theta_{\max} \) changes and the rotation is exactly \( r/2 \).) Using, our dynamics, we see that

\[
\begin{align*}
\Delta \theta(\text{now} + t_r) &= \Delta \theta(\text{now}) + 2ct_r \\
&\leq \Delta \theta(\text{now}) + 2c(r/2) \\
&\leq \Delta \theta(\text{now}) + cr \leq \Delta \theta(\text{now}^-)
\end{align*}
\]

where \( \Delta \theta(\text{now}^-) \) is \( \Delta \theta \) the instant before a new observation is taken. Thus, under any execution including a turn, \( \Delta \theta \) decreases if \( \epsilon < 1 \).

The quantity \( \Delta \theta \) does not change under any other actions. So, by Inductive Invariance, the Lemma holds.

**Lemma 6.** \( \dot{\theta}_d \pm \frac{\Delta \theta}{2} \) is bounded away from \( \pm \frac{\pi}{2} \).

**Proof:** We prove this lemma by contradiction. Consider the case where \( \dot{\theta}_d + \Delta \theta/2 \geq \frac{\pi}{2} \). Since \( \frac{\Delta \theta(t)}{2} \leq \frac{\Delta \theta(0)}{2} \leq \frac{\pi}{2} \), this implies \( \dot{\theta}_d > 0 \). The quantity \( \dot{\theta}_d \) increases only when we have a rotation. We know that for a rotation of size \( r/2 \), we have reduced the size of \( \Delta \theta \) by at least \( r \). Thus, if \( \theta'_d \) was the value of \( \dot{\theta}_d \) before rotation, \( \theta'_d + r/2 + \frac{\Delta \theta - r}{2} \geq \frac{\pi}{2} \).

This implies either \( \theta'_d > 0 \) or \( \Delta \theta/2 \geq \frac{\pi}{2} \). If we continue this process the total number of rotations we have performed (until \( \theta'_d = 0 \)), this implies that \( \Delta \theta/2 \geq \frac{\pi}{2} \), which is a contraction.
The same analysis holds for the case where $\theta_d - \Delta \theta/2 \leq \frac{\pi}{2}$. Thus, the lemma holds.

**Lemma 7.** For sufficiently large $x_1$, $\Delta \theta/2(x_1) \leq \arctan \left( \frac{2\delta}{x_1} \right)$.

**Proof:** We show this lemma using a geometric argument. Given an arbitrary $\Delta \theta/2 < \Delta \theta(0)/2$, $\Delta x/2 < \delta$, $\theta_d$, and $t_y$ specified by the system, we know that

$$\sin(\Delta \theta(t)/2) \geq \frac{(\Delta x_2(t)/2 + \delta) \cos \theta_d(t)}{kt_y}$$

(23)

for sufficiently large $k \in \mathbb{Z}^+$ and this implies that eventually the size of $\Delta \theta$ will get smaller. This is because as the set of reachable $\theta$ causes the size of the reachable size $x_2$ to the point where the distance between minimum and maximum values is greater than our sensor resolution $\delta$. At this time, we will definitely be able to reduce our uncertainty in $x_2$ and by extension our uncertainty in $\theta$. Now, consider Equation 23 initialized from our starting position, letting the observations and robot movements be the worst-case for reduction of uncertainty in $\theta$. This corresponds to the case where $y_2(i,t_y) = 0$ for all $i \in \{0, 1, \ldots, k\}$. For all sufficiently large $k \in \mathbb{Z}^+$,

$$\sin(\Delta \theta(0)/2 - r) = \frac{2\delta}{kt_y}$$

(24)

The term $r$ corresponds to the amount $\Delta \theta/2$ is not reduced to give us equality, i.e. $\Delta \theta(0)/2 - r = \Delta \theta(kt_y)$.

We will demonstrate that this is the worst-case scenario by bounding all other cases. Suppose at time $t_1 = (k-1)t_y$ we have some nonzero $\theta_d = \frac{\pi}{2(k-1)}$. Noticing that we start with $\theta_d = 0$ and that for any rotation of size $a$, the system reduces $\Delta \theta$ by at least $2\alpha$, this implies the inequality

$$\sin(\Delta \theta(0)/2 - r) \leq \frac{\Delta x_2(t_1)/2 + \delta}{kt_y} \cos \left[ \frac{\pi t - r}{2} \right]$$

(25)

Showing that

$$\left( \Delta x_2(t_1)/2 + \delta \right) \cos \left[ \frac{\pi t - r}{2} \right] \leq 2\delta$$

(26)

is sufficient to demonstrate Equation 24 represents the worst case. Equation eq:uncertaintyreducesufficient holds for any $r \in [0, \Delta \theta/2]$ as $\Delta x_2/2 \leq \delta$ at all $kt_y$ due to the observation.

Thus, a coarse bound on $\Delta \theta(kt_y)/2$ is

$$\Delta \theta(kt_y)/2 \leq \arcsin \left( \frac{2\delta}{kt_y} \right)$$

(27)

We know moving distance $kt_y$ implies we’ve moved to at least $x_1 = kt_y \cos(\Delta \theta(0)/2)$ so

$$\Delta \theta/2(x_1) \leq \arcsin \left( \frac{2\delta}{x_1} \cos(\Delta \theta(0)/2) \right)$$

(28)

or we can use the tangent explicitly

$$\Delta \theta/2(x_1) \leq \arctan \left( \frac{2\delta}{x_1} \right)$$

(29)

Thus, we’ve proved the lemma.

**Lemma 8.** $x_{2,\max}, x_{2,\min}$ are bounded with respect to $x^d, x_1$, and $\delta$.

**Proof:** If the robot moves upward away from the line $x_2 = 0$, the uncertainty in angle $\Delta \theta$ will be reduced and it will rotate so that its estimated direction of travel makes angle $\theta_d$ with the line $x_2 = 0$. If $\hat{x}_2$ gets large enough then $\theta_d + \frac{\Delta \theta}{2} \leq 0$. In this case, $x_2$ cannot grow any larger. This event takes place any time that

$$\frac{\Delta \theta}{2} \leq \theta_d = \arctan \left( \frac{\hat{x}_2}{x^d - x_1} \right)$$

(30)

Assuming the worst-case estimate for continuous variables $(x_1, x_2)$ and using the bound on $\Delta \theta/2$ from Lemma IV-B,

$$\frac{\Delta \theta}{2} \leq \arctan \left( \frac{2\delta}{x_1} \right) \leq \theta_d = \arctan \left( \frac{x_2 - \delta \sqrt{2}}{x^d - x_1 + \delta \sqrt{2}} \right)$$

(31)

This implies the condition holds when

$$\left( 2\delta x^d - 2\delta^2 \right) \frac{1}{x_1} - \left( 2\delta - \frac{\delta}{\sqrt{2}} \right) \leq x_2$$

(32)

Since this condition holds true only at sample times, we know that $x_2$ will not increase beyond this value plus $t_y$. Also, we must consider a lower limit due to sensing. In fact, we cannot bound $x_2$ below $\delta$. Thus, our bound for any time now is

$$x_2 \leq \min \left\{ \left( 2\delta x^d - 2\delta^2 \right) \frac{1}{x_1} - \left( 2\delta - \frac{\delta}{\sqrt{2}} \right), \delta \right\} + t_y$$

(33)

A symmetric argument holds for $x_{2,\min}$ so the lemma holds.

We have now bounded $x_2$ to a tube around the line $x_2 = 0$. Using this bound in conjunction with the fact that the reachability of the system is contained inside the system with a policy that rotates, i.e.

$$|x_2(x_1)| \leq |x_{2,\max}(0) + x_1 \sin(\Delta \theta(0)), x_{2,\min}(0) - x_1 \sin(\Delta \theta(0))|$$

(34)

we can construct a tube whose maximum value occurs at the intersection of these curves. See Figure 8 for some example bounding sets. We can use this property along with the other lemmas we have proved to the following proposition.

**Prop 2.** The robot will eventually reach the waypoint, i.e. $x^d - x_1 \leq \delta$, remaining in a bounded set around the ray from $(0, 0)$ to $(0, x^d)$, and will have $\Delta \theta(t_{final}) \leq \Delta \theta(0)$. $\Delta \theta(t_{final}) \leq \arctan \left( \frac{2\delta}{x^d} \right)$ if $x^d$ is sufficiently large.

**Proof:** Lemma 6 implies $x_1$ makes progress on every step so after a finite number of steps we will eventually reach $x^d - \delta$. By Lemma 8 combined with Equation 34 we can bound $x_2$ around zero. From Lemma we see that if $x^d$ is sufficiently large then we can reduce $\Delta \theta(t_{final})$ from $\Delta \theta(0)$.

Now that we have shown that the robot is guaranteed to reach a single waypoint, we can reason about a sequence of waypoints. The analysis between waypoints we have already done will hold for every pair of waypoints if the assumptions are met.
Prop 3. For a sequence of \( n \) waypoints, if the computed tube bounding trajectories between any two waypoints does not intersect any obstacles when the starting state is place within the reachable box around each waypoint one of the following conditions holds,

1. \( 2\epsilon\pi(n+1) + \Delta\theta^0(0) < \pi \)
2. Define \( \Delta\theta_0 = \Delta\theta(\text{now} = 0) \) and if for every \( 0 < i < n \),
   \[
   \Delta\theta_{i+1} \leq \min \left\{ \Delta\theta_i, \tan^{-1} \left( \frac{2\delta}{d_i - \delta} \right) \right\} + 2\epsilon(\phi_i + \pi/2) < \pi
   \]

where \( d_i, \phi_i \) is the distance, rotation between waypoints \( i - 1 \) and \( i \)

then the system will reach \( X_{\text{goal}} \).

Proof: We have demonstrated that the robot can move between any two waypoints. We just need to ensure rotations from one waypoint move to the next do not cause the uncertainty in \( \theta \) to increase beyond \( \pi \). Conditions (1) and (2) both ensure this, (1) holding in cases where there is a lot of slack
and (2) where we need to reason about the reduction of \( \Delta\theta \) moving along the line between waypoints. At the terminal point of any trajectory from one waypoint to another, the robot may end with state in \( x_1 \in \{ x^d \pm \delta \} \) and \( x_2 \) bounded by either Lemma 8 or Equation 34. This creates a starting condition for the next waypoint. If we draw a circle whose radius is the maximum of the bound on \( x_1 \) and \( x_2 \) around each waypoint, this is a superset of the reachable ending points for the trajectory moving to this waypoint. If we start the reachability tube between the next two waypoints at every point on the circumference of that circle and end it at the waypoint, take the union of those sets, and fill in any gaps to make the set compact, this is an over-approximation of the set of reachable points in the continuous space. We can then collision check this set with obstacles to guarantee reachability.

V. Conclusions

In this paper, we considered verification of the property that a control policy, combined with a specific robot system in a known environment, will lead to collision-free paths that always reach a goal. We first discussed the difficulty of the Navigation Problem and why we turn to state feedback policies. We then analyzed two robot systems: a robot whose continuous state is in \( \mathbb{R}^2 \) with rectangular constraints and a mobile robot whose continuous state is in \( SE(2) \) whose policy is a series of waypoints.

Although no surprising results were uncovered in either analysis, this work gives us a valuable proof of concept of using analytic methods from the theory of verification of hybrid systems to build simpler verification systems for some classes of robotic systems. It seems that using these methods, even the simplest systems will require a large amount of analysis. However, in the cases considered, once the analysis was complete we were left with surprisingly simple conditions for verifying obstacle avoidance and reachability of the goal set.

One future topic of research is applying these techniques to more complicated systems. One system we have partially analyzed is a Rhino robot arm, a system used in the ECE 470 Instructional Laboratory. The robot modeled is a five link manipulator with four revolute joints and its task is to pick up and place blocks in a workspace observed by an external camera. It would be useful to compute the reachable set of the end effector and the links of the robot so that guarantees that obstacles can be avoided in the workspace under a sequence of waypoint commands can be made, even after a large error has built up in the joint encoders after many moves. This would also allow us to see if a sequence of moves is guaranteed to be close enough to the target block to pick up or if we may need to consider a different policy.

The work of this paper is a stepping stone towards combining heuristic policies with verification to avoid computing optimal state feedback policies. Because of the computational intractability of computing optimal state feedback policies for systems with nontrivial dynamics and high dimensionality of the state vector, we’d like to approximate optimal policies or generate policies by heuristic and use methods like those we’ve explored to check the viability of our policies.

For the systems discussed in this paper, naively combining a policy construction heuristic with the verification rules discussed may yield preliminary results. However, this project has shown several interesting new directions for a more cohesive scheme. It seems likely that we can develop algorithms that use results of these types of analysis to efficiently synthesize or modify policies. Thus, rather than computing a policy and then verifying, we could attempt to construct heuristic policies that
automatically satisfy the conditions laid out in the analysis. In this way, our policies would be verified by construction.

Another interesting direction is analyzing the sensitivities to variations in sensor uncertainty, sample times, process uncertainty, and other parameters of the model. An unexpected result of the first analysis was quantitative bounds on the minimum sizes of passages and goal regions that the robot could be guaranteed to handle, with respect to sensor and process uncertainty. By varying the parameters of the model, we could find places on the continuum of parameters where a small change produces a qualitatively different result. For example, if we decrease the sensor uncertainty then perhaps new passages between obstacles will be opened up that we can safely navigate. This type of analysis could have a wide variety of applications.

In conclusion, the exploration done for this project has been interesting. Although we analyzed only some simple systems, it was enough for significant insight into the problem. As a result, significant possibility for many extensions exist.

REFERENCES