UNIVERSITY OF ILLINOIS AT URBANA-CHAMPAIGN

Reachability Analysis of Mobile Robot Trajectories in Polygons with SpaceEx

Alli Nilles

December 5, 2017

Overall Goals

- model a system, related to billiards, which has applications in robotics

Overall Goals

- model a system, related to billiards, which has applications in robotics
- become more familiar with reachability analysis and how to state queries in terms of invariants

Overall Goals

- model a system, related to billiards, which has applications in robotics
- become more familiar with reachability analysis and how to state queries in terms of invariants
- result: created an interface from an existing simulator for this system to SpaceEx

Blind, Bouncing Robots[1]

Model the robot as a point moving **in straight lines** in the plane, "bouncing" off the boundary at a **fixed angle** θ from the normal:

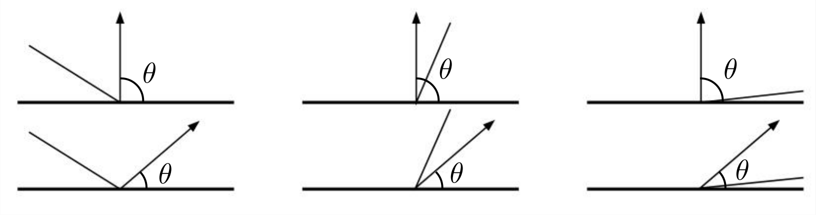


Figure 1. A point robot moving in the plane. The top row shows bounces at zero degrees from the normal. The second row shows bounces at 50 degrees clockwise from normal.

[1](Erickson and LaValle 2013)

Questions from Robotics

- What kind of tasks are robots with extremely simple control laws capable of performing?

Questions from Robotics

- What kind of tasks are robots with extremely simple control laws capable of performing?
- *Localization:* Will the robot become "trapped" in a small part of the environment? Or a low-dimensional motion pattern?

Questions from Robotics

- What kind of tasks are robots with extremely simple control laws capable of performing?
- *Localization:* Will the robot become “trapped” in a small part of the environment? Or a low-dimensional motion pattern?
- *Coverage:* How completely and evenly does the robot cover the environment?

Questions from Robotics

- What kind of tasks are robots with extremely simple control laws capable of performing?
- *Localization:* Will the robot become "trapped" in a small part of the environment? Or a low-dimensional motion pattern?
- *Coverage:* How completely and evenly does the robot cover the environment?

Questions from Robotics

- What kind of tasks are robots with extremely simple control laws capable of performing?
- *Localization:* Will the robot become "trapped" in a small part of the environment? Or a low-dimensional motion pattern?
- *Coverage:* How completely and evenly does the robot cover the environment?

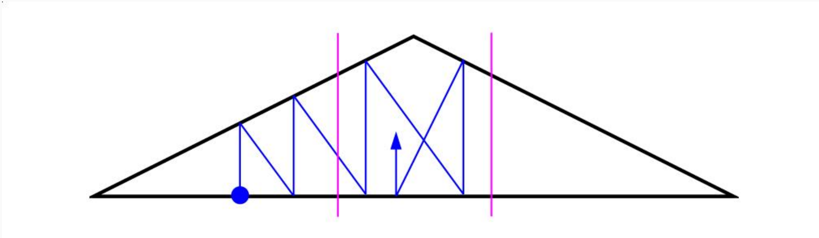


Figure 2. In this environment, bouncing at the normal, the robot will become trapped in the area between the purple lines.

Implementation

- Assume we know environment exactly

Implementation

- Assume we know environment exactly
- Can implement on a roomba with bump sensor and IR prox detector[2]

[2] (Lewis and O'Kane 2013), IJRR

Implementation

- Assume we know environment exactly
- Can implement on a roomba with bump sensor and IR prox detector[2]
- “Collisions” can be virtual - for example, robot w/ camera stops when it is collinear with two landmarks, and rotates until one landmark is at a certain heading

[2](Lewis and O’Kane 2013), IJRR

Implementation

- Assume we know environment exactly
- Can implement on a roomba with bump sensor and IR prox detector[2]
- "Collisions" can be virtual - for example, robot w/ camera stops when it is collinear with two landmarks, and rotates until one landmark is at a certain heading
- Also useful model of very small "robots" or microorganisms,[3] or robots in low-bandwith environments

[2](Lewis and O'Kane 2013), IJRR
[3](Spagnolie et al. 2017)

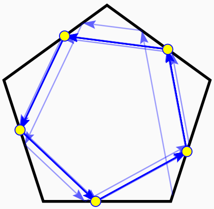
Applications

NASA’s Mars Roomba Begins Mission To Clean Dust From Planet’s Surface

According to NASA, the Mars Roomba’s edge-cleaning mode will allow the vehicle to scour even the crevices where mountains meet the planet’s surface.

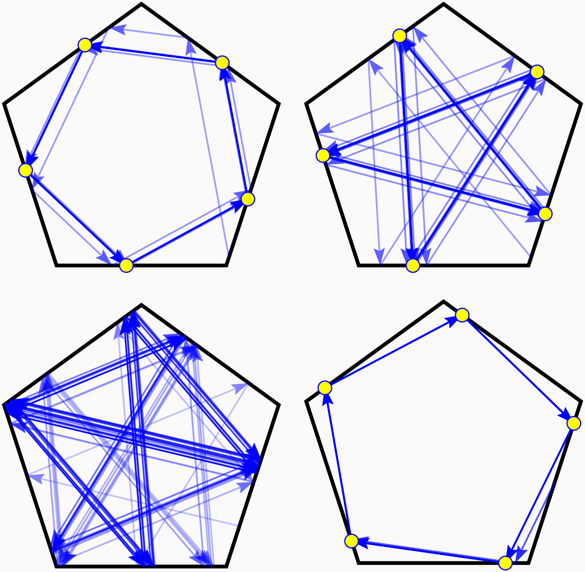
Discovery Through Simulation

- Haskell with *Diagrams* library (Yorgey 2012)
- fixed-angle bouncing, specular bouncing, add noise
- render diagrams from simulations automatically[4]

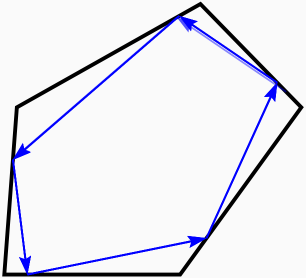


[4] `https://github.com/alexandroid000/bounce`

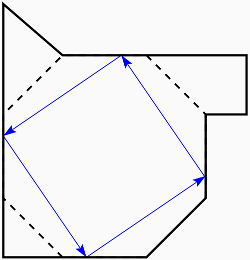
Simulation Results



Other Polygons



(a) A stable orbit in a sheared pentagon.



(b) A stable orbit in a nonconvex environment.

Figure 4. Stable orbits also exist in non-regular polygons.

What We Want

- check if cycle exists where theory says it does for convex polygons

What We Want

- check if cycle exists where theory says it does for convex polygons
- do experiments with nonconvex polygons to develop theory there

What We Want

- check if cycle exists where theory says it does for convex polygons
- do experiments with nonconvex polygons to develop theory there
- minimize simulation / discretization / floating point artifacts

What We Want

- check if cycle exists where theory says it does for convex polygons
- do experiments with nonconvex polygons to develop theory there
- minimize simulation / discretization / floating point artifacts
- extend to multiple robots and nondeterministic bouncing

What We Want

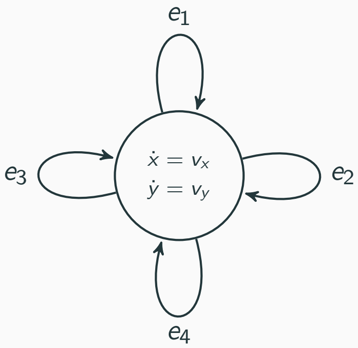
- check if cycle exists where theory says it does for convex polygons
- do experiments with nonconvex polygons to develop theory there
- minimize simulation / discretization / floating point artifacts
- extend to multiple robots and nondeterministic bouncing

What We Want

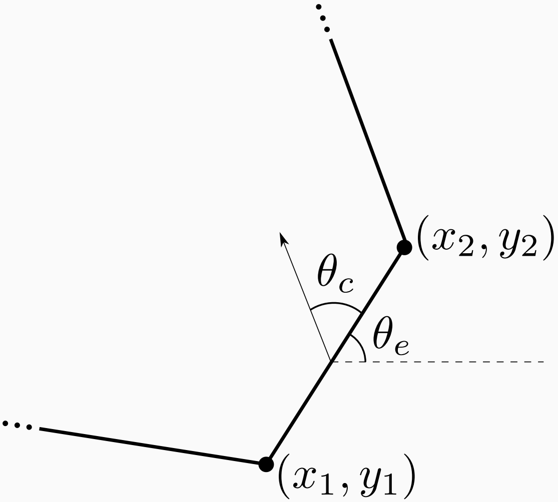
- check if cycle exists where theory says it does for convex polygons
- do experiments with nonconvex polygons to develop theory there
- minimize simulation / discretization / floating point artifacts
- extend to multiple robots and nondeterministic bouncing

Tools from this class especially help with the last two

Reachability Modelling Approach



Modelling Transitions



Modelling Transitions

If robot is colliding with wall,

$(x, y) = (x_1, y_1) + s((x_2, y_2) - (x_1, y_1))$, and $0 \leq s < 1$

Modelling Transitions

If robot is colliding with wall,

$(x, y) = (x_1, y_1) + s((x_2, y_2) - (x_1, y_1))$, and $0 \leq s < 1$

Pre: $\frac{x-x_1}{x_2-x_1} == \frac{y-y_1}{y_2-y_1} \wedge 0 \leq s < 1$

Note 1: This decides "corner collisions" consistently.

Note 2: Requires special case for vertical/horizontal edges

Modelling Transitions

If robot is colliding with wall,

$(x, y) = (x_1, y_1) + s((x_2, y_2) - (x_1, y_1))$, and $0 \leq s < 1$

Pre: $\frac{x - x_1}{x_2 - x_1} == \frac{y - y_1}{y_2 - y_1} \wedge 0 \leq s < 1$

Note 1: This decides "corner collisions" consistently.

Note 2: Requires special case for vertical/horizontal edges

Eff:

$v_x := \cos(\theta_{edge} + \theta_c)$

$v_y := \sin(\theta_{edge} + \theta_c)$

Modelling Transitions

If robot is colliding with wall,

$(x, y) = (x_1, y_1) + s((x_2, y_2) - (x_1, y_1))$, and $0 \leq s < 1$

Pre: $\frac{x - x_1}{x_2 - x_1} == \frac{y - y_1}{y_2 - y_1} \wedge 0 \leq s < 1$

Note 1: This decides "corner collisions" consistently.

Note 2: Requires special case for vertical/horizontal edges

Eff:

$v_x := \cos(\theta_{edge} + \theta_c)$

$v_y := \sin(\theta_{edge} + \theta_c)$

Compute concretely in Haskell, put floating point numbers into SpaceEx... Motivation for using dReach?

Modelling Transitions

If robot is colliding with wall,

$(x, y) = (x_1, y_1) + s((x_2, y_2) - (x_1, y_1))$, and $0 \leq s < 1$

Pre: $\frac{x-x_1}{x_2-x_1} == \frac{y-y_1}{y_2-y_1} \wedge 0 \leq s < 1$

Note 1: This decides "corner collisions" consistently.

Note 2: Requires special case for vertical/horizontal edges

Eff:

$v_x := \cos(\theta_{edge} + \theta_c)$

$v_y := \sin(\theta_{edge} + \theta_c)$

Compute concretely in Haskell, put floating point numbers into SpaceEx... Motivation for using dReach?

Only supports single-valued θ_c (for now)

Results

```
loc1 = Location 1 "interior"
        "-500.0 &lt;= x &amp;&amp;
         x &lt;= 0.0 &amp;&amp;
         0 &lt;= y &amp;&amp;
         y &lt;= 500"
        "x'==vx &amp; y'==vy"

square_ha :: HA
square_ha = HA   { name = "test"
                 , params = mkParams $ mkPoly sq
                 , locations = [loc1]
                 , transitions = mkTs (mkPoly sq) (0 @@ rad)
                 }
```

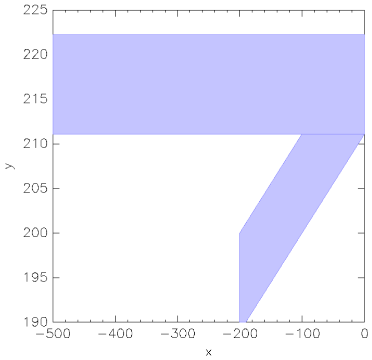
Results

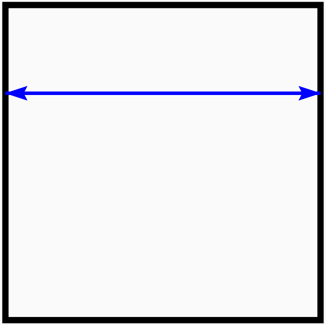

```xml
<?xml version="1.0" encoding="iso-8859-1"?>
<sspaceex xmlns="http://www-verimag.imag.fr/xml-namespaces/sspaceex"version="0.2" math="SpaceEx">
	<component id="test">
	<param name="x" type="real" local="true" d1="1" d2="1" dynamics="any" />
	<param name="y" type="real" local="true" d1="1" d2="1" dynamics="any" />
	<param name="vx" type="real" local="true" d1="1" d2="1" dynamics="const" />
	<param name="vy" type="real" local="true" d1="1" d2="1" dynamics="const" />
	<param name="e1" type="label" local="false" />
	<param name="e2" type="label" local="false" />
	<param name="e3" type="label" local="false" />
	<param name="e4" type="label" local="false" />
	<location id="1" name="interior">
		<invariant>-500.0 &lt;= x &amp;&amp; x &lt;= 0.0 &amp;&amp; 0 &lt;= y &amp;&amp; y &lt;= 500</invariant>
		<flow>x'==vx &amp; y'==vy</flow>
	</location>
	<transition source="1" target="1" asap="true" >
		<label>e1</label>
		<guard>x - (0.0) &lt; (0.001) &amp;&amp; x - (0.0) &gt; -(0.001) &amp;&amp; (0.0) &lt;= y &amp;&amp; y &lt; (500.0)</guard>
		<assignment>vx := (-1.0) &amp; vy := (-0.000000000000000049789962505147994)</assignment>
	</transition>
	<transition source="1" target="1" asap="true" >
		<label>e2</label>
		<guard>y - (500.0) &lt; (0.001) &amp;&amp; y - (500.0) &gt; -(0.001) &amp;&amp; (-500.00000000000006) &lt;= x &amp;&amp; x &lt;
(-0.00000000000000005551115123125783)</guard>
		<assignment>vx := (-0.00000000000000006123233995736766) &amp; vy := (-1.0)</assignment>
	</transition>
	<transition source="1" target="1" asap="true" >
		<label>e3</label>
		<guard>x - (-500.00000000000006) &lt; (0.001) &amp;&amp; x - (-500.00000000000006) &gt; -(0.001) &amp;&amp; (0.0) &lt;= y &amp;&amp; y &lt;
		<assignment>vx := (1.0) &amp; vy := (-0.00000000000000001749191776789837)</assignment>
	</transition>
	<transition source="1" target="1" asap="true" >
		<label>e4</label>
		<guard>y - (0.0) &lt; (0.001) &amp;&amp; y - (0.0) &gt; -(0.001) &amp;&amp; (-500.0000000000001) &lt;= x &amp;&amp; x &lt; (0.0)</guard>
		<assignment>vx := (0.00000000000000006123233995736766) &amp; vy := (1.0)</assignment>
	</transition>
</component>
</sspaceex>
```

Results of Simulations

When bouncing between parallel sides, SpaceEx finds fixed point within a few iterations!

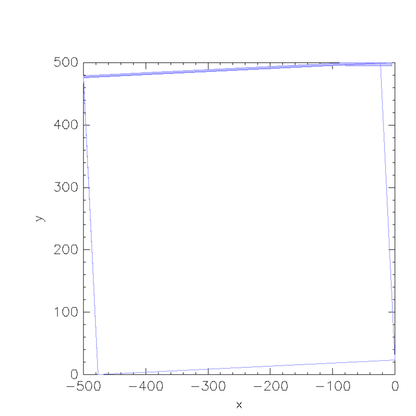
This type of bouncing is geometrically exact: $f_{1,3}(f_{3,1}(x)) = x$ if $f_{i,j}$ is the mapping from side e_i to side e_j.

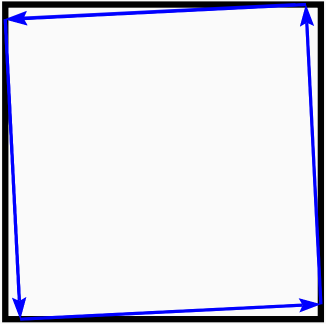




Results of Simulations - Nonconvergence w/ Asymptotic Stability

When periodic orbit is asymptotically stable, SpaceEx does not appear to converge (700+ iterations, several minutes, how long to wait?)





Limitations / Future Work

- `asap` transitions not working for some reason: must specify invariant in location to keep robot from “escaping” the polygon

Limitations / Future Work

- `asap` transitions not working for some reason: must specify invariant in location to keep robot from “escaping” the polygon
- Also not sure how to specify nonconvex invariants

Limitations / Future Work

- `asap` transitions not working for some reason: must specify invariant in location to keep robot from "escaping" the polygon
- Also not sure how to specify nonconvex invariants
- Both supposed to be included in PHAVer on SpaceEx (Minopoli and Frehse 2014)

Limitations / Future Work

- `asap` transitions not working for some reason: must specify invariant in location to keep robot from "escaping" the polygon
- Also not sure how to specify nonconvex invariants
- Both supposed to be included in PHAVer on SpaceEx (Minopoli and Frehse 2014)

Limitations / Future Work

- `asap` transitions not working for some reason: must specify invariant in location to keep robot from "escaping" the polygon
- Also not sure how to specify nonconvex invariants
- Both supposed to be included in PHAVer on SpaceEx (Minopoli and Frehse 2014)

- Naive code generation: chunking strings together (only a few fields in xml file we need to change)

Limitations / Future Work

- `asap` transitions not working for some reason: must specify invariant in location to keep robot from "escaping" the polygon
- Also not sure how to specify nonconvex invariants
- Both supposed to be included in PHAVer on SpaceEx (Minopoli and Frehse 2014)

- Naive code generation: chunking strings together (only a few fields in xml file we need to change)
- Next step is to use Haskell XML Toolbox (HXT) to make this less janky

More Future Work

- Build in support for bounce angle intervals

More Future Work

- Build in support for bounce angle intervals
- Limit cycle detection with reachability (if robot starts in interval on edge i, show it will not reach the complement of that interval)

More Future Work

- Build in support for bounce angle intervals
- Limit cycle detection with reachability (if robot starts in interval on edge i, show it will not reach the complement of that interval)
- Use this as subroutine for synthesis algorithms: given environment, what bounce angles produce paths with certain properties (coverage, limit cycles)?

More Future Work

- Build in support for bounce angle intervals
- Limit cycle detection with reachability (if robot starts in interval on edge i, show it will not reach the complement of that interval)
- Use this as subroutine for synthesis algorithms: given environment, what bounce angles produce paths with certain properties (coverage, limit cycles)?
- Modelling / synthesizing strategies over multiple angles (generate multiple automata and compose)

More Future Work

- Build in support for bounce angle intervals
- Limit cycle detection with reachability (if robot starts in interval on edge i, show it will not reach the complement of that interval)
- Use this as subroutine for synthesis algorithms: given environment, what bounce angles produce paths with certain properties (coverage, limit cycles)?
- Modelling / synthesizing strategies over multiple angles (generate multiple automata and compose)
- Multiple robots and collisions (including robots sticking together?)

References

Erickson, L. H., and S. M. LaValle. 2013. "Toward the Design and Analysis of Blind, Bouncing Robots." In *IEEE International Conference on Robotics and Automation*.

Lewis, Jeremy S., and Jason M. O'Kane. 2013. "Planning for Provably Reliable Navigation Using an Unreliable, Nearly Sensorless Robot." *International Journal of Robotics Research* 32 (11): 1339–54.

Minopoli, Stefano, and Goran Frehse. 2014. "Non-Convex Invariants and Urgency Conditions on Linear Hybrid Automata." In *International Conference on Formal Modeling and Analysis of Timed Systems*, 176–90. Springer.

Spagnolie, Saverio E, Colin Wahl, Joseph Lukasik, and Jean-Luc Thiffeault. 2017. "Microorganism Billiards." *Physica D: Nonlinear Phenomena* 341. Elsevier: 33–44.

Yorgey, Brent A. 2012. "Monoids: Theme and Variations (Functional Pearl)." In *ACM SIGPLAN Notices*, 47:105–16. 12. ACM.