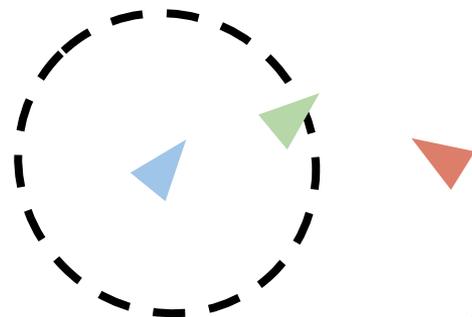
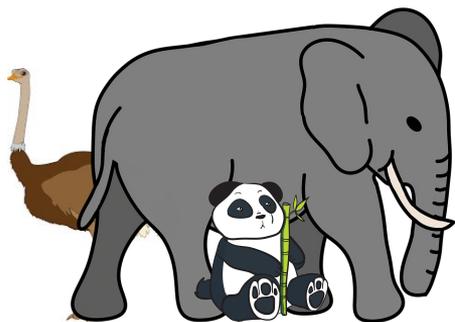




Modeling an Adversarial Poacher-Ranger Hybrid Game

Maia Iyer and Benjamin Gilby

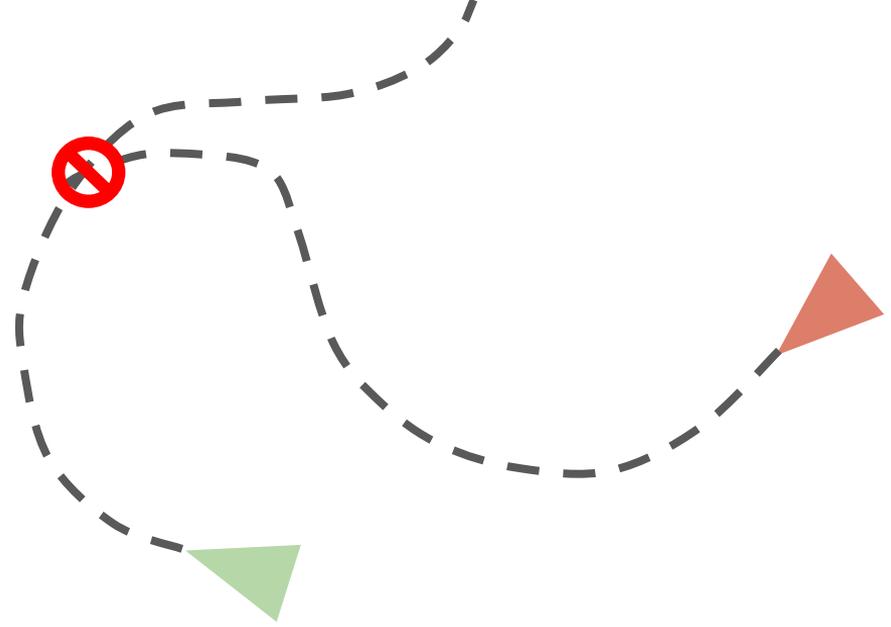
maiai@andrew.cmu.edu bgilby@andrew.cmu.edu



Introduction

Pursuit-evasion games:

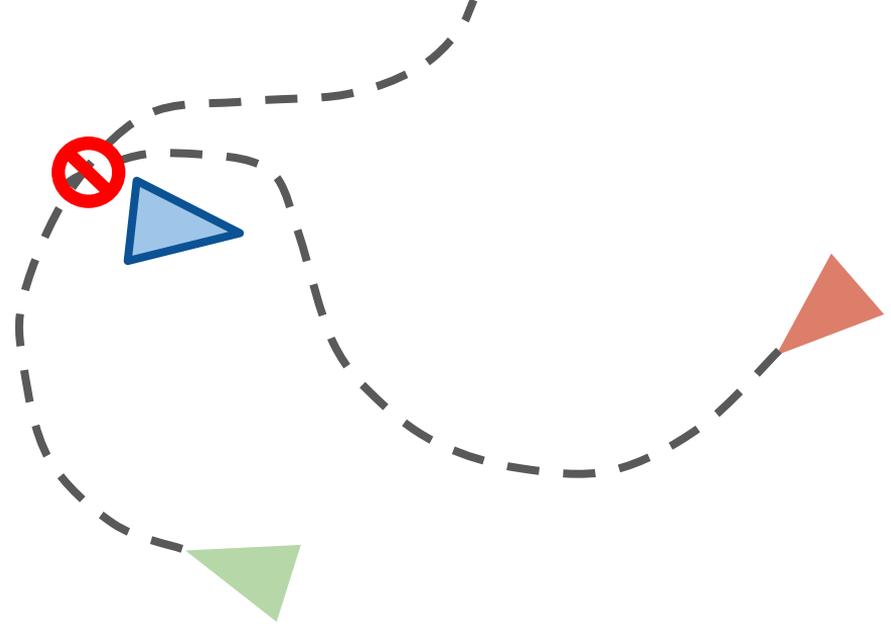
- Two-player
- Real-dimensional space
- Win-condition defined by positions in space



Introduction

Active-Target Defense Differential Games:

- Two-player, **three entities**

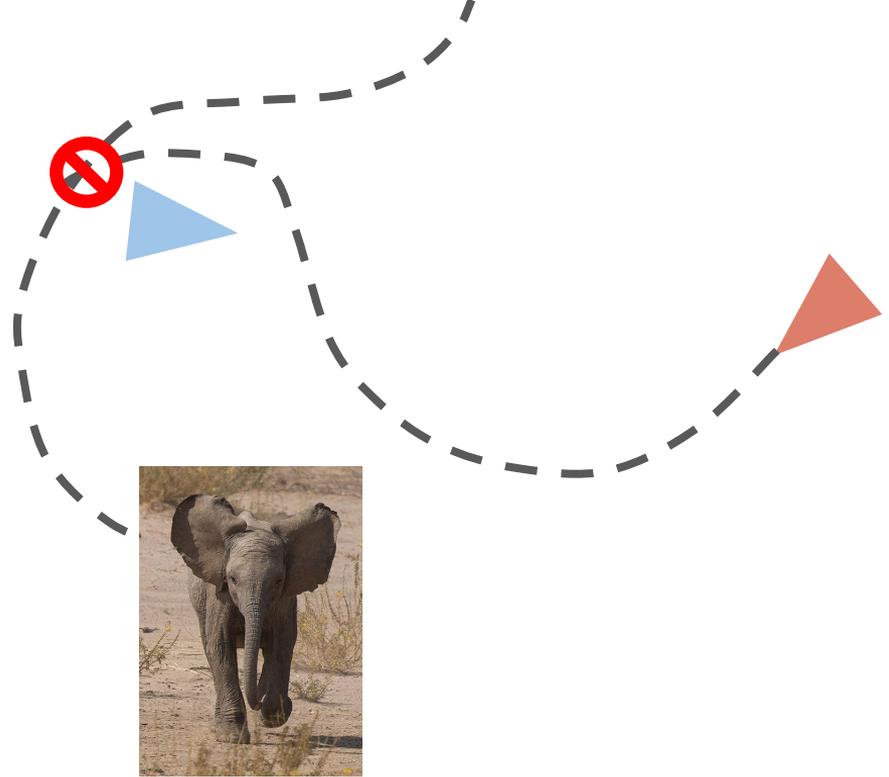


Previous Work

Perfect-information (in the literature)

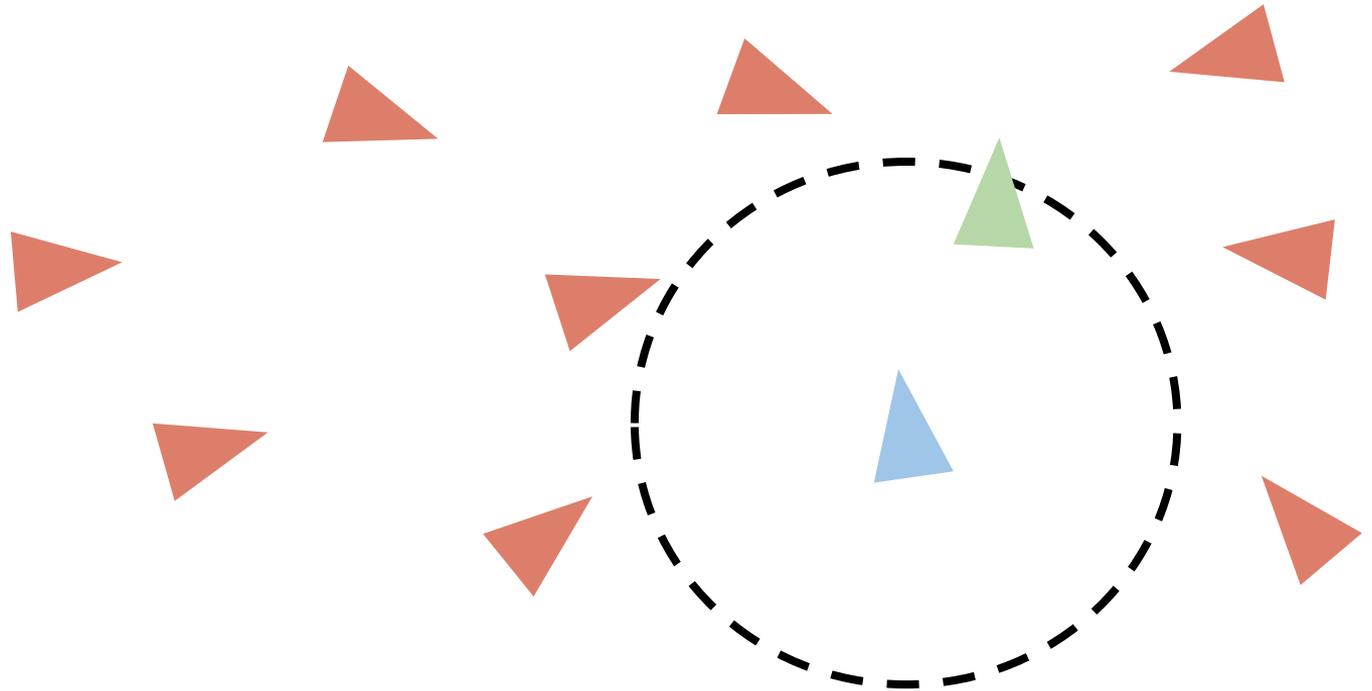
Nash Equilibrium

- Optimal play
- Probabilistic strategies



Introduction

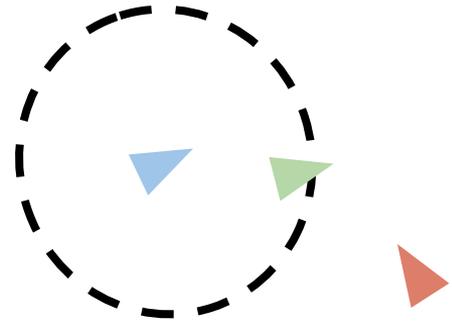
Partially-Observable Active-Target Defense Differential Games (**POATDDG**)



Basic Poacher-Ranger Game

Partially-Observable Active-Target Defense Differential Games (**POATDDG**)

- 3 entities: poacher (attacker), ranger (defender), prey (target)
- Ranger/Prey win if:
 - ranger catches poacher or if poacher is never able to kill prey
- Poacher wins if it reaches prey position and not in ranger's range



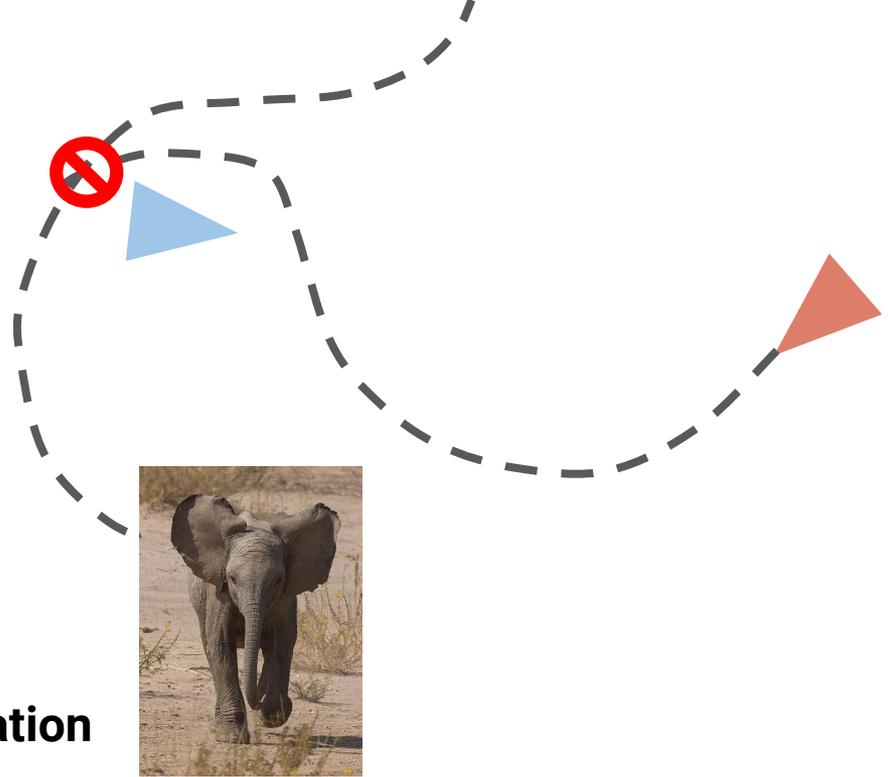
Introduction

Hybrid systems:

- Added **discrete control** dynamic
- Can **augment motion fidelity**

Hybrid games:

- Can model the **adversarial aspect**
- Can directly capture **imperfect information**

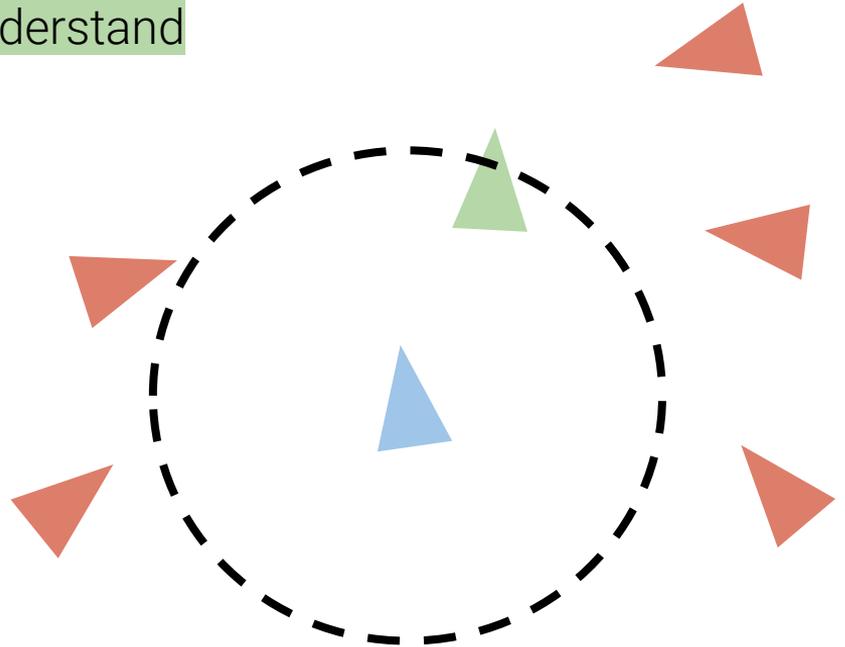


How can we incorporate
asymmetric partial information
into the game using the
hybrid game framework to
reason about winning ranger
strategies?

Contributions

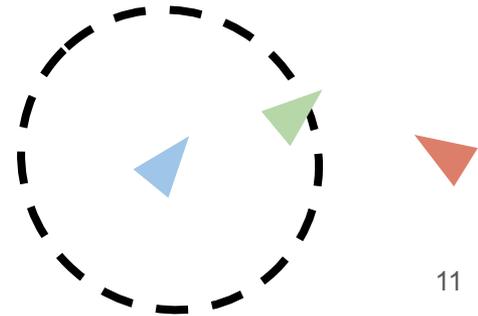
Incorporating Partial Information

- **State Observability** is ability to **observe the environment**
- **Structural Observability** is ability to understand win conditions and **play optimally**



Model Description - Constants

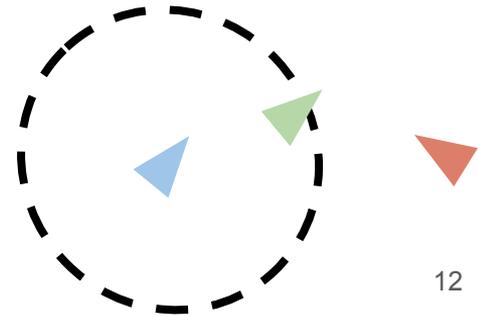
```
Definitions /* CONSTANTS */  
  Real maxVp; /* max poacher velocity */  
  Real maxVr; /* max ranger velocity */  
  Real maxVt; /* max target velocity */  
  
  Real killr; /* how close poacher should be to kill target */  
  Real capr; /* how close ranger should be to catch poacher */  
End.
```



Model Description - Players

We focus on ranger strategy with imperfect cooperation by prey

- Prey and Poacher are on the same team
- Ranger is demonic (needs existence of one path of strategies)
- Prey/Poacher are angelic (ranger needs to work on all possible prey/poacher paths)

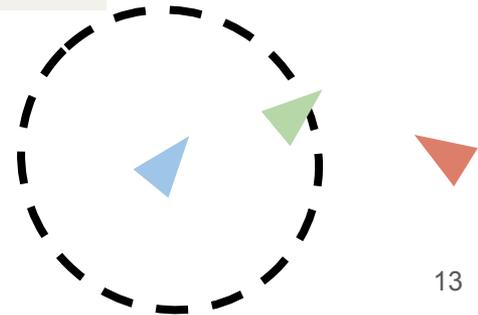


Model Description - Hybrid Game Structure

Problem

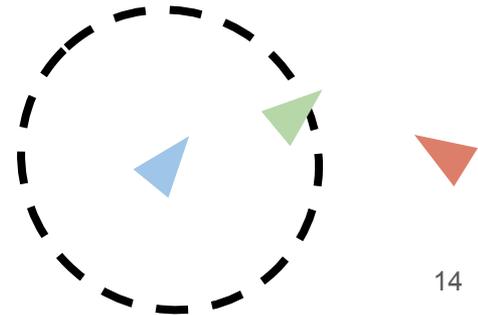
```
/* CONSTANT assumptions */  
->  
[  
  /* Initial positions chosen by all entities */  
  {  
    /* Control */  
    /* Dynamics */  
    /* poacher responsible to ensure not caught */  
    ?(distancesq(xP, xR, yP, yR) > capr^2);  
  }* /* poacher can choose to continue the game */  
]( /* Safety condition */  
  distancesq(xP, xT, yP, yT) > killr^2 /* target never caught */  
)
```

End.



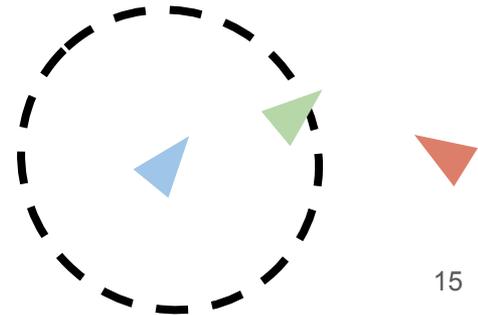
Model Description - Dynamics

```
/* Dynamics */
{pos' = v &
  /* Poacher caught -> ranger win */
  distancesq(xP, xR, yP, yR) >= capr^2 &
  /* Target caught -> poacher win */
  distancesq(xP, xT, yP, yT) >= killr^2 &
};
/* poacher responsible to ensure not caught */
?(distancesq(xP, xR, yP, yR) > capr^2);
```



Model Description - Control

```
/* Target Control */  
dxT1 := *; dyT1 := *; ?(dxT1^2 + dyT1^2 <= maxVt^2);  
/* Ranger Control, generally predefined */  
{dxR1 := *; dyR1 := *; ?(dxR1^2 + dyR1^2 <= maxVr^2);}^@;  
/* Poacher Control */  
dxP1 := *; dyP1 := *; ?(dxP1^2 + dyP1^2 <= maxVp^2);
```



Model Description - Partial State Observability

```
3 Definitions
4  /* Hybrid Programs */
5  /* this strategy keeps ranger at rest */
6  HP rangerStay ::= {dxR1:=0; dyR1 :=0;};
7  /* this strategy assumes ranger observes prey velocity and copies*/
8  HP rangerGo ::= {dxR1:=dxT1; dyR1:=dyT1};
9 End.
10
11 Problem
12  /* Initial conditions & constant assumptions */ ( )
13  ->
14  [
15    /* Initial positions */
16    {
17      /* Target Control */
18      dxT1 := *; dyT1 := *; ?(dxT1^2 + dyT1^2 <= maxVt^2);
19      /* Ranger Control, generally predefined */
20      {rangerGo; ?(dxR1^2 + dyR1^2 <= maxVr^2);}^@;
21      /* Poacher Control */
22      dxP1 := *; dyP1 := *; ?(dxP1^2 + dyP1^2 <= maxVp^2);
23      /* Dynamics */
24    }*
25  ]( /* Safety condition */ )
26 End.
```

Model Description - Assumptions

- Bounded space: we focused on making prey faster
- Players can choose initial state: we focused on existence of mid-game strat
- $\text{killr}=0$

Three POATDDGs - Extending the Poacher-Ranger Game

- **Panda Game**

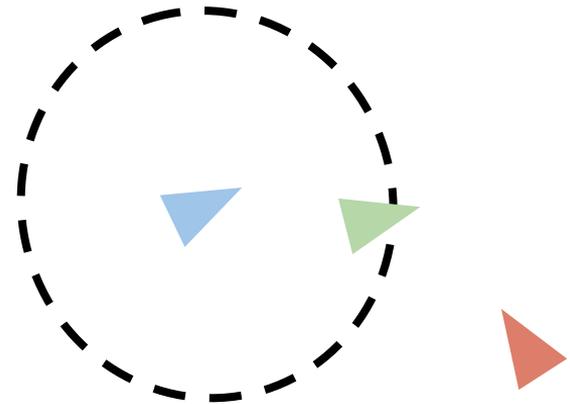
- Stationary prey, unknown poacher parameters

- Elephant Game

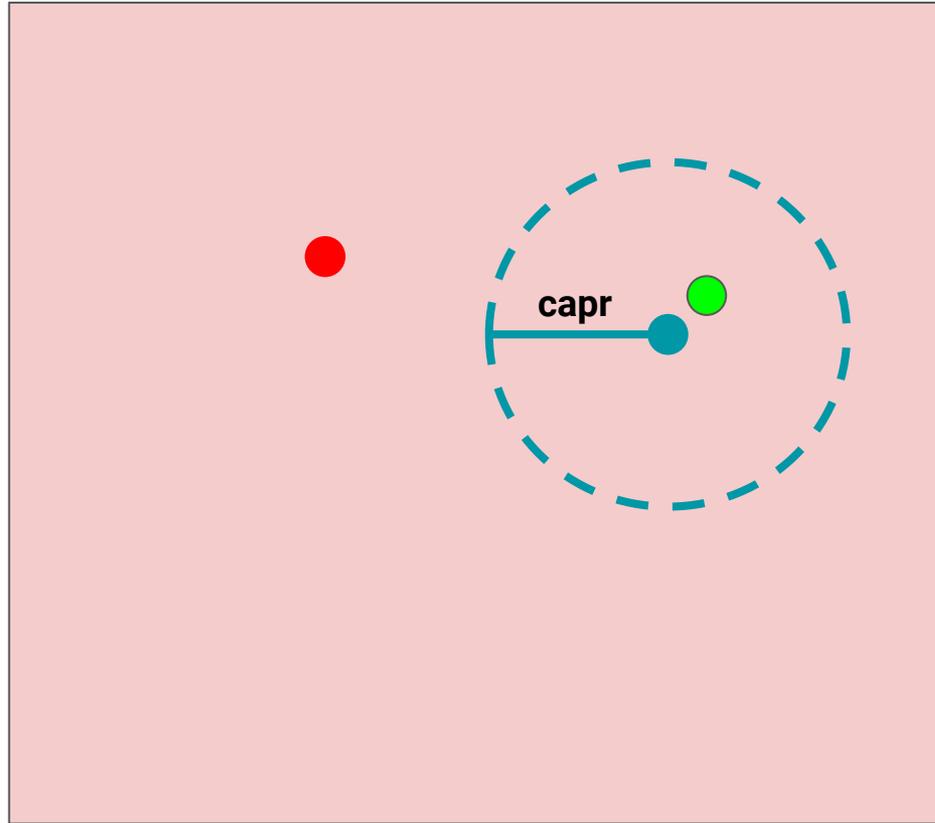
- Relatively slow moving prey, unknown poacher parameters

- Ostrich Game

- Relatively fast moving prey, unknown poacher parameters
- Prey freezes when poacher is near (within "prey" distance)
- Ranger knows when prey sees poacher

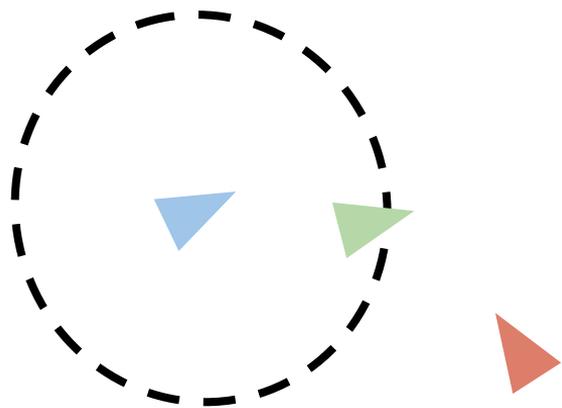


Panda Game



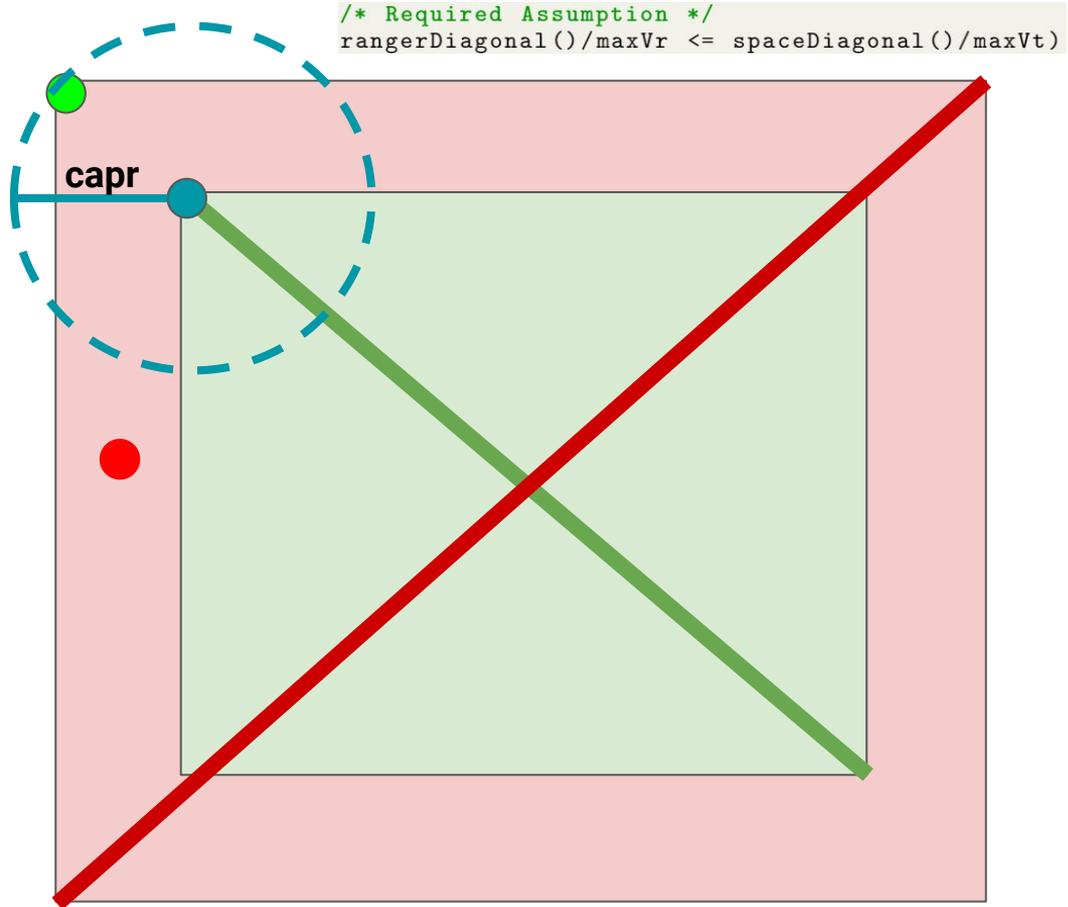
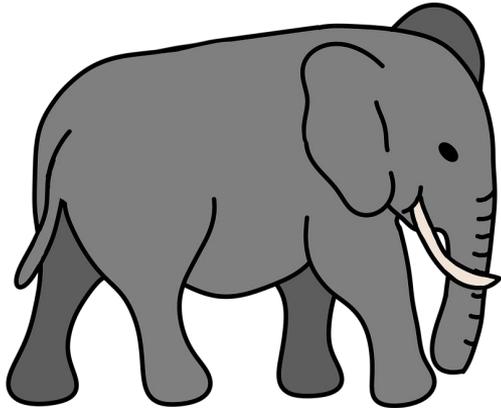
Three POATDDGs - Extending the Poacher-Ranger Game

- Panda Game
 - Stationary prey, unknown poacher parameters
- **Elephant Game**
 - Relatively slow moving prey, unknown poacher parameters
- Ostrich Game
 - Relatively fast moving prey, unknown poacher parameters
 - Prey freezes when poacher is near (within "prey" distance)
 - Ranger knows when prey sees poacher



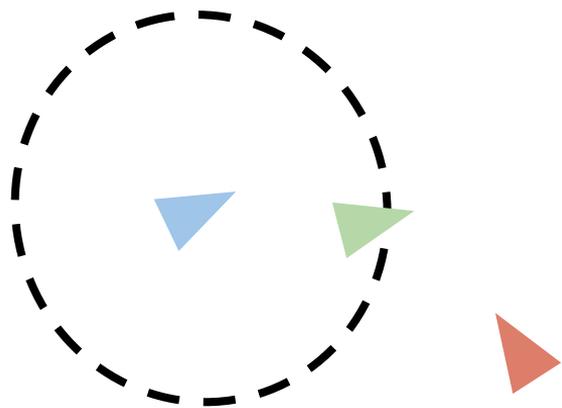
Elephant Game

- Green rectangle = subspace
- Red diagonal = spaceDiagonal
- Green diagonal = rangerDiagonal



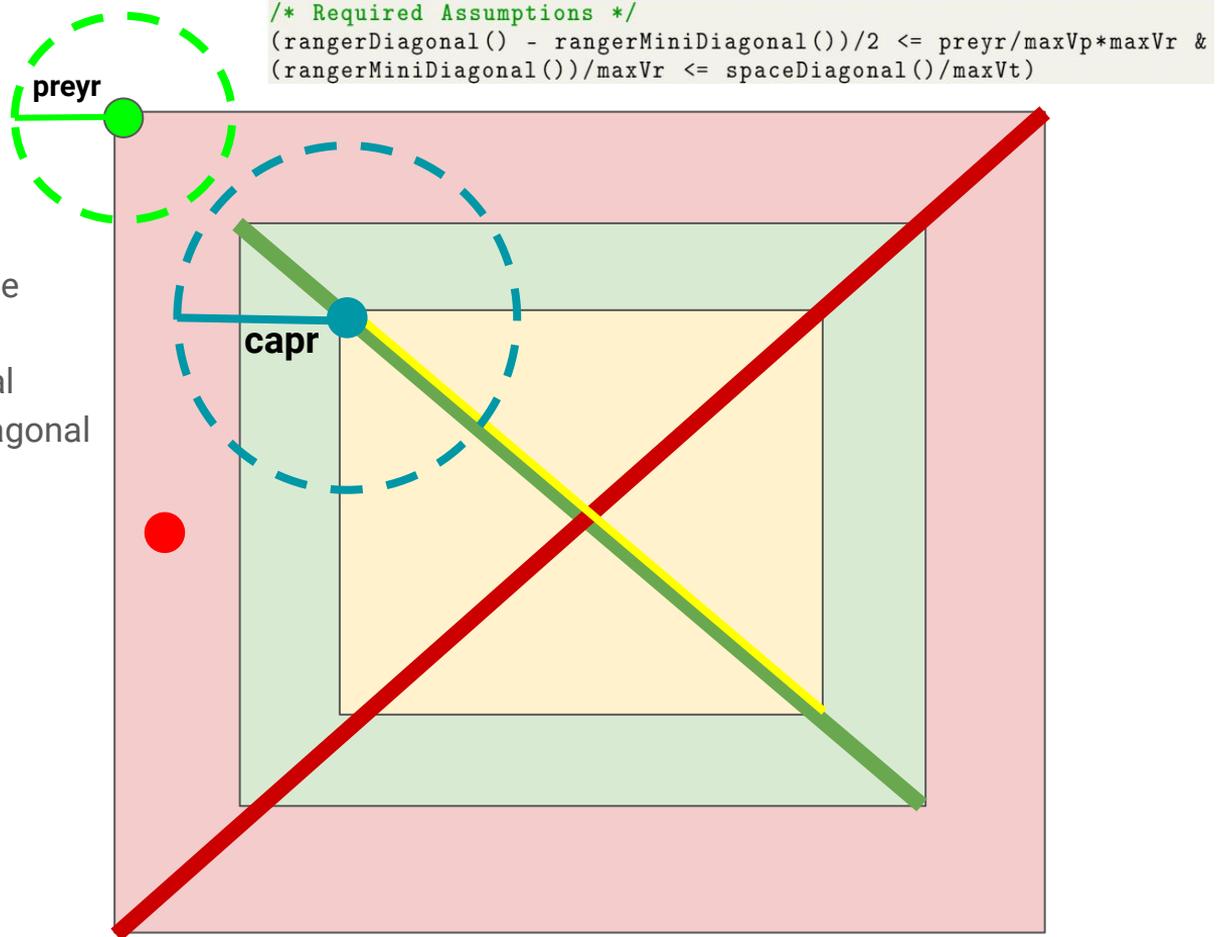
Three POATDDGs - Extending the Poacher-Ranger Game

- Panda Game
 - Stationary prey, unknown poacher parameters
- Elephant Game
 - Relatively slow moving prey, unknown poacher parameters
- **Ostrich Game**
 - Relatively fast moving prey, unknown poacher parameters
 - Prey freezes when poacher is near (within “prey” distance)
 - Ranger knows when prey sees poacher



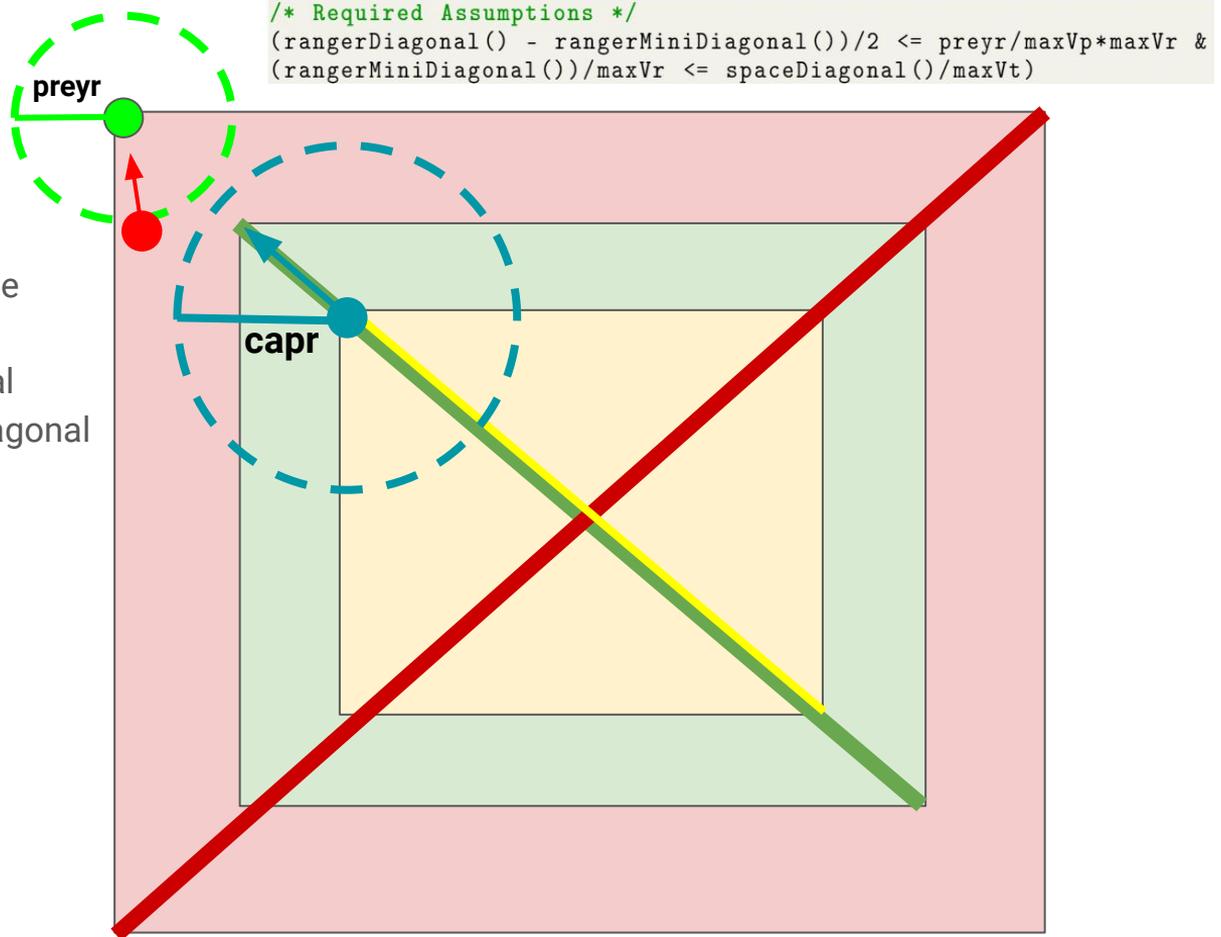
Ostrich Game

- Green rectangle = subspace
- Yellow rectangle = miniSubspace
- Red diagonal = spaceDiagonal
- Green diagonal = rangerDiagonal
- Yellow diagonal = rangerMiniDiagonal



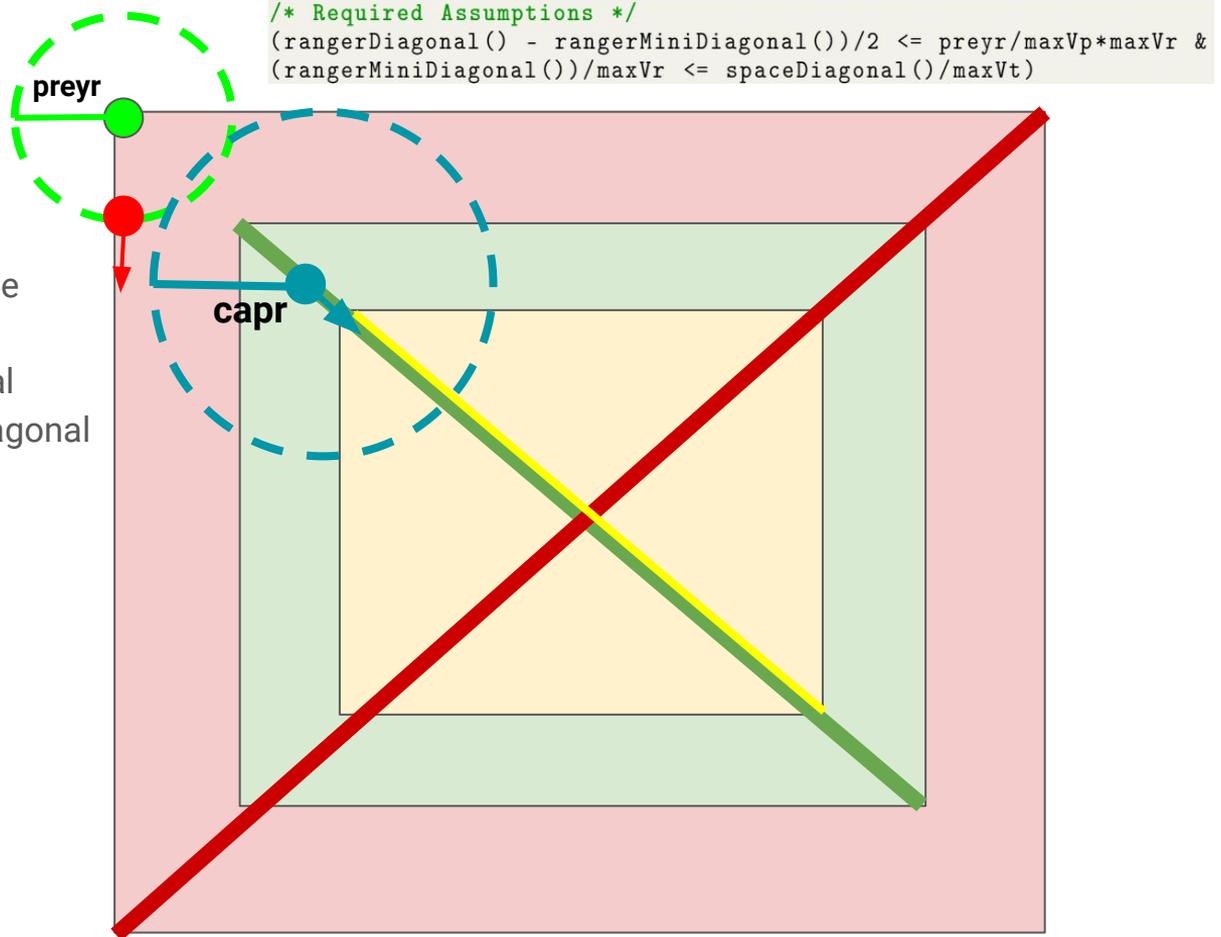
Ostrich Game

- Green rectangle = subspace
- Yellow rectangle = miniSubspace
- Red diagonal = spaceDiagonal
- Green diagonal = rangerDiagonal
- Yellow diagonal = rangerMiniDiagonal



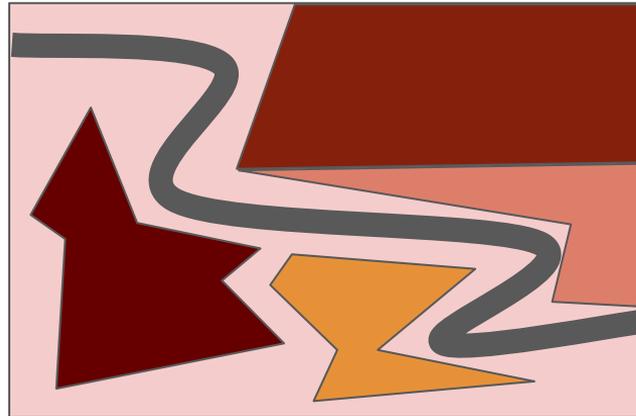
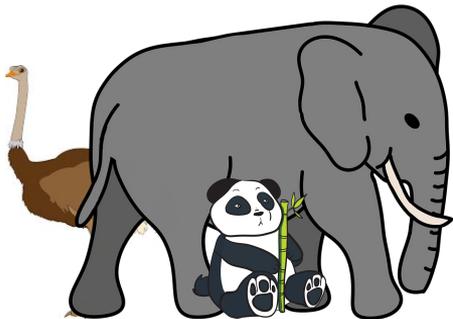
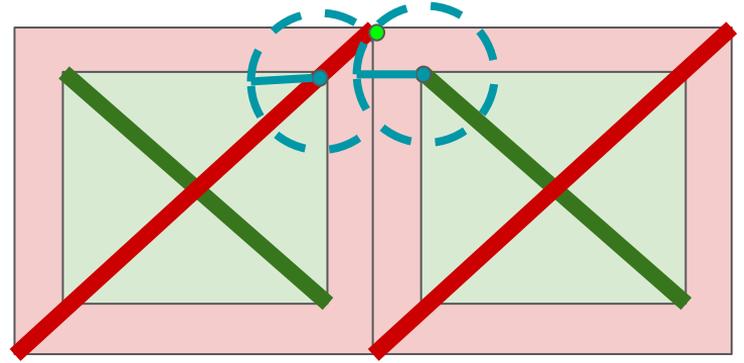
Ostrich Game

- Green rectangle = subspace
- Yellow rectangle = miniSubspace
- Red diagonal = spaceDiagonal
- Green diagonal = rangerDiagonal
- Yellow diagonal = rangerMiniDiagonal



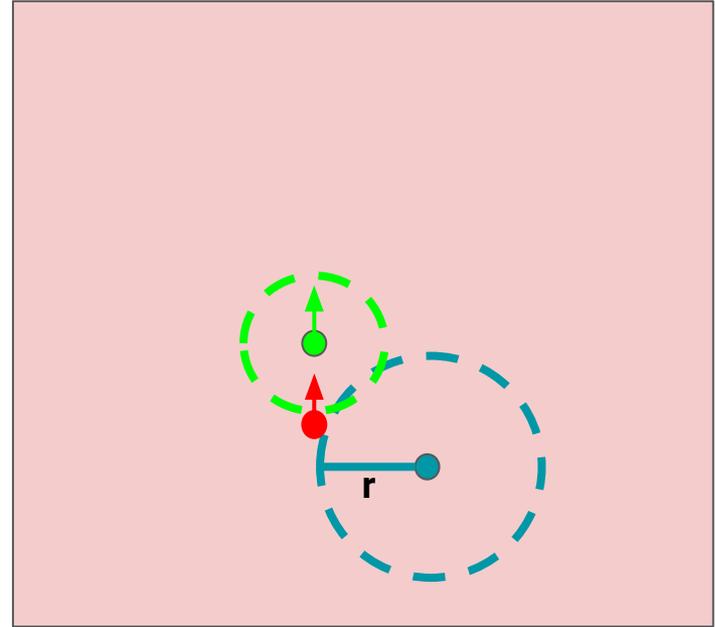
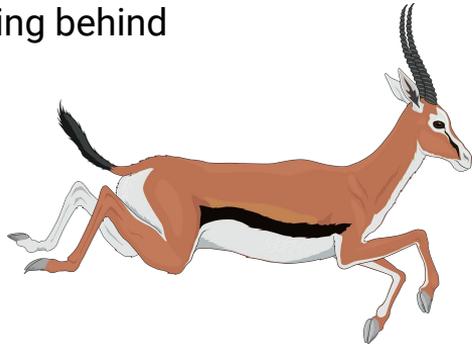
Future Work

- Extensions to Multiple Entities
 - Double the rangers -> double the playing space
- New Environmental Factors
 - Terrain
 - Acceleration



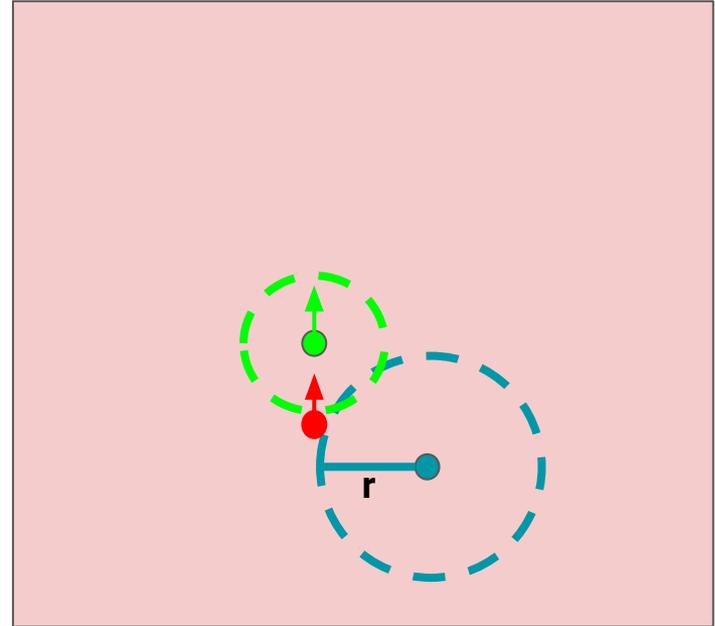
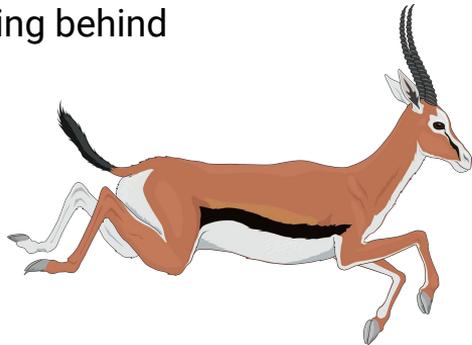
Challenges

- Formulating and balancing the game
- Strategy Complexity
 - Translating a geometric strategy in a dGL proof
- Gazelle Game
 - Prey flees from poacher
 - Complex strategies
 - Falling behind



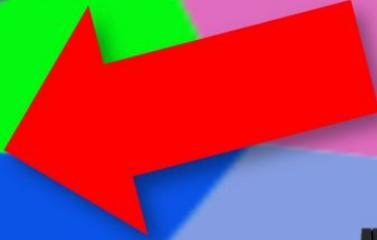
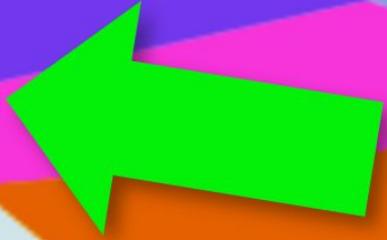
Challenges

- Formulating and balancing the game
- Strategy Complexity
 - Translating a geometric strategy in a dGL proof
- Gazelle Game
 - Prey flees from poacher
 - Complex strategies
 - Falling behind



References

- [1] André Platzer. Differential game logic. *ACM Trans. Comput. Log.*, 17(1):1:1–1:51, 2015.
- [2] André Platzer. *Logical Foundations of Cyber-Physical Systems*. Springer, Cham, 2018.
- [3] Rufus Isaacs. Differential games: Their scope, nature, and future. *J. Optim. Theory Appl.*, 3(5):283–295, May 1969.
- [4] Isaac E. Weintraub, Meir Pachter, and Eloy Garcia. An introduction to pursuit-evasion differential games, 2020.
- [5] F. Fang, T. Nguyen, Arunesh Sinha, Shahrzad Gholami, Andrew Plumptre, L. Joppa, Milind Tambe, Margaret Driciru, F. Wanyama, Rob Critchlow, and Colin Beale. Predicting poaching for wildlife protection. *IBM Journal of Research and Development*, 61:3:1–3:12, 11 2017.
- [6] Timothy Kuiper, Blessing Kavhu, Nobesuthu A. Ngwenya, Roseline Mandisodza-Chikerema, and E.J. Milner-Gulland. Rangers and modellers collaborate to build and evaluate spatial models of african elephant poaching. *Biological Conservation*, 243:108486, 2020.
- [7] Y.-C Ho, A.E. Bryson, and S. Baron. Differential games and optimal pursuit-evasion strategies. *Automatic Control, IEEE Transactions on*, AC10:385 – 389, 11 1965.
- [8] Efstathios Bakolas and Panagiotis Tsiotras. Optimal pursuit of moving targets using dynamic voronoi diagrams. pages 7431 – 7436, 01 2011.



```

] (distancesq(xP1, xT1, yP1, yT1) > killr^2 &
  (tSinceTargetSeenPoacher >= preyr/maxVp -> (xR1 = spaceToMiniRanger(xT1, maxX) & yR1 = spaceToMiniRanger(yT1,
maxY))) &
  (xR1 = ((tSeeingPoacher/(preyr/maxVp))*spaceToRanger(xT1, maxX) + (1-(tSeeingPoacher/(preyr
/maxVp)))*spaceToMiniRanger(xT1, maxX))) &
  (yR1 = ((tSeeingPoacher/(preyr/maxVp))*spaceToRanger(yT1, maxY) + (1-(tSeeingPoacher/(preyr
/maxVp)))*spaceToMiniRanger(yT1, maxY))) &
  (tSeeingPoacher <= preyr/maxVp - tSinceTargetSeenPoacher) &
  (0 <= tSeeingPoacher & tSeeingPoacher <= preyr/maxVp) &
  (0 <= tSinceTargetSeenPoacher & tSinceTargetSeenPoacher <= preyr/maxVp) &
  (tSinceTargetSeenPoacher >= preyr/maxVp -> !targetSeesPoacher(xP1, yP1, xT1, yT1)) &
  inBorder(xR1, yR1) & inBorder(xP1, yP1) & inBorder(xT1, yT1)
  & (targetSeesPoacher(xP1, yP1, xT1, yT1)->tSinceTargetSeenPoacher = 0)
End.

```