TRACKING MOVING OBJECTS WITH FEW HANDOVERS

David Eppstein
Michael Goodrich
Maarten Löffler
University of California, Irvine

INGREDIENTS

INGREDIENTS

- 1 Plane

INGREDIENTS

- 1 Plane

INGREDIENTS

- 1 Plane
- 1 Unpredictable Moving Object

INGREDIENTS

- 1 Plane
- 1 Unpredictable Moving Object

INGREDIENTS

- 1 Plane
- 1 Unpredictable Moving Object
  - Travels unpredictably in the plane

INGREDIENTS

- 1 Plane
- 1 Unpredictable Moving Object
  - Travels unpredictably in the plane

INGREDIENTS

- 1 Plane
- 1 Unpredictable Moving Object
  - Travels unpredictably in the plane

INGREDIENTS

- 1 Plane
- 1 Unpredictable Moving Object
  - Travels unpredictably in the plane

INGREDIENTS

- 1 Plane
- 1 Unpredictable Moving Object
  - Travels unpredictably in the plane

INGREDIENTS

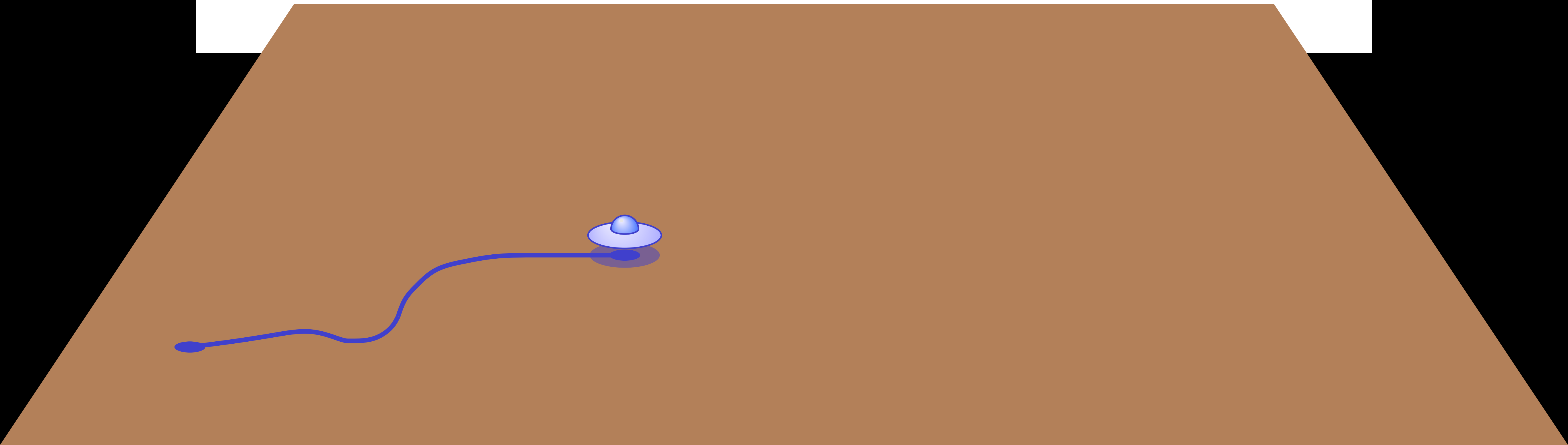
- 1 Plane
- 1 Unpredictable Moving Object
  - Travels unpredictably in the plane

INGREDIENTS

- 1 Plane
- 1 Unpredictable Moving Object
  - Travels unpredictably in the plane

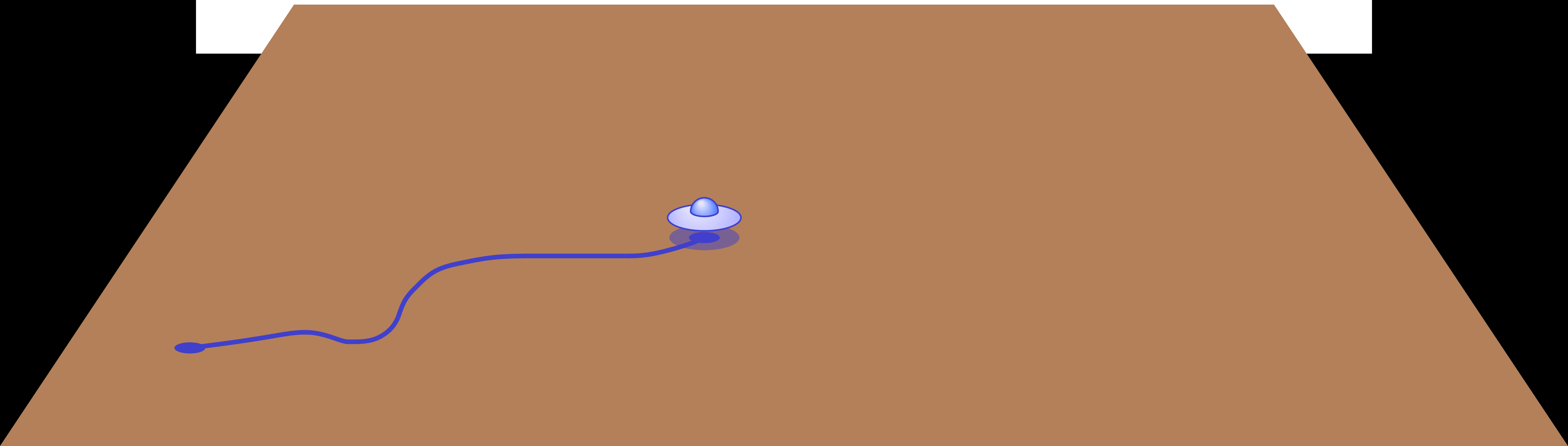
INGREDIENTS

- 1 Plane
- 1 Unpredictable Moving Object
  - Travels unpredictably in the plane



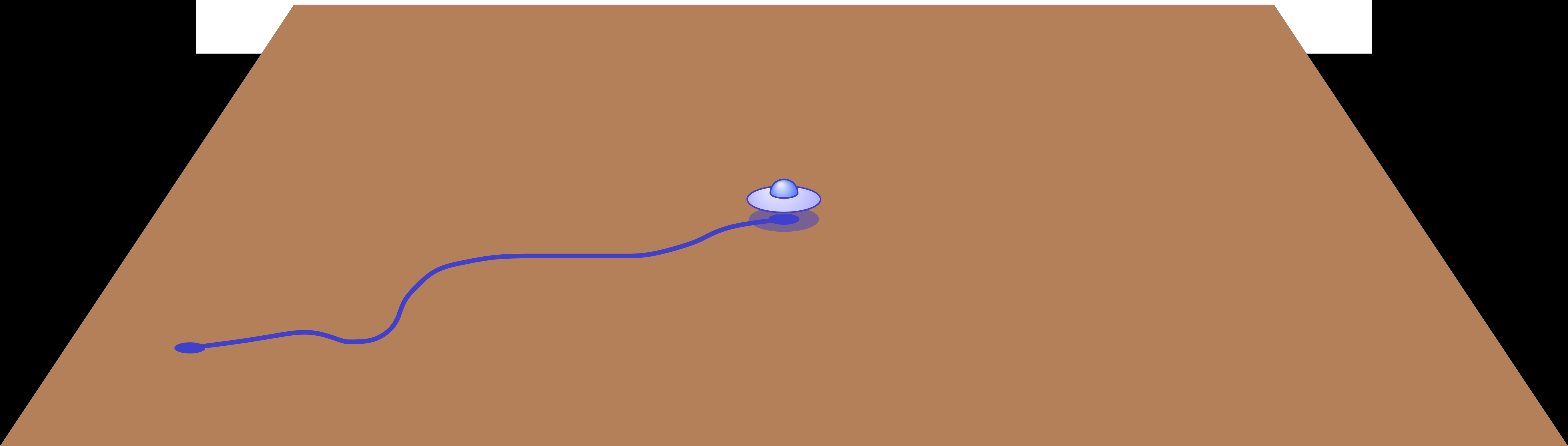
INGREDIENTS

- 1 Plane
- 1 Unpredictable Moving Object
  - Travels unpredictably in the plane



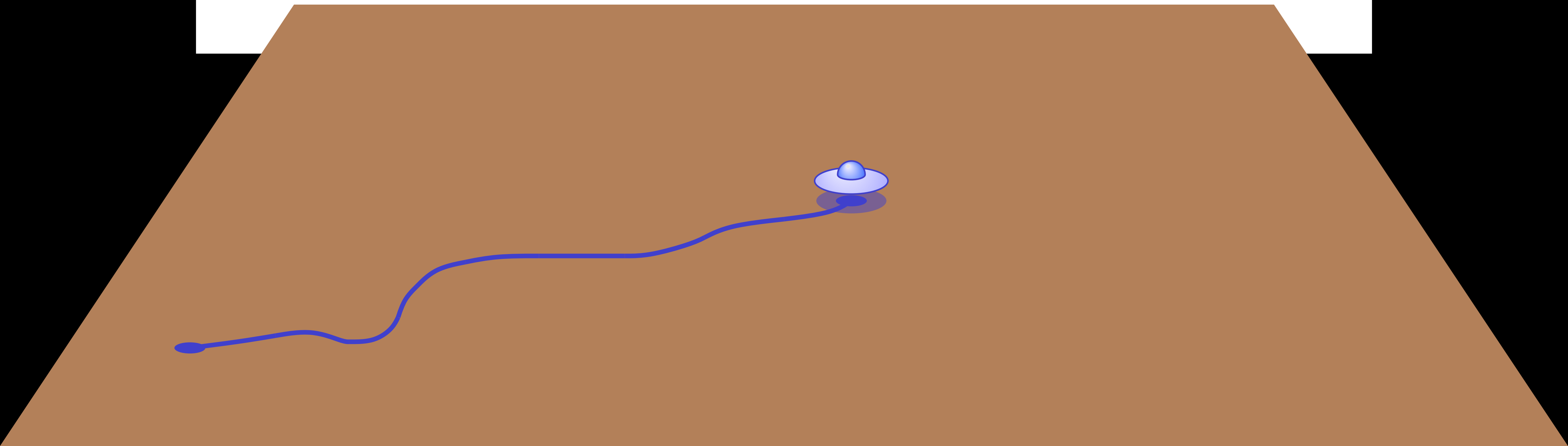
INGREDIENTS

- 1 Plane
- 1 Unpredictable Moving Object
  - Travels unpredictably in the plane



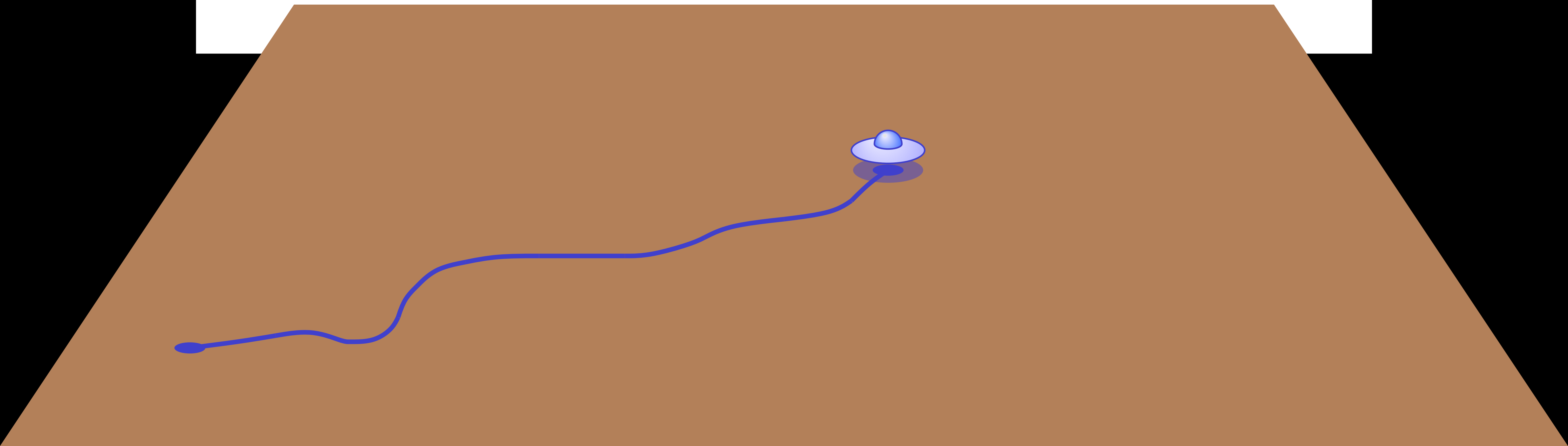
INGREDIENTS

- 1 Plane
- 1 Unpredictable Moving Object
  - Travels unpredictably in the plane



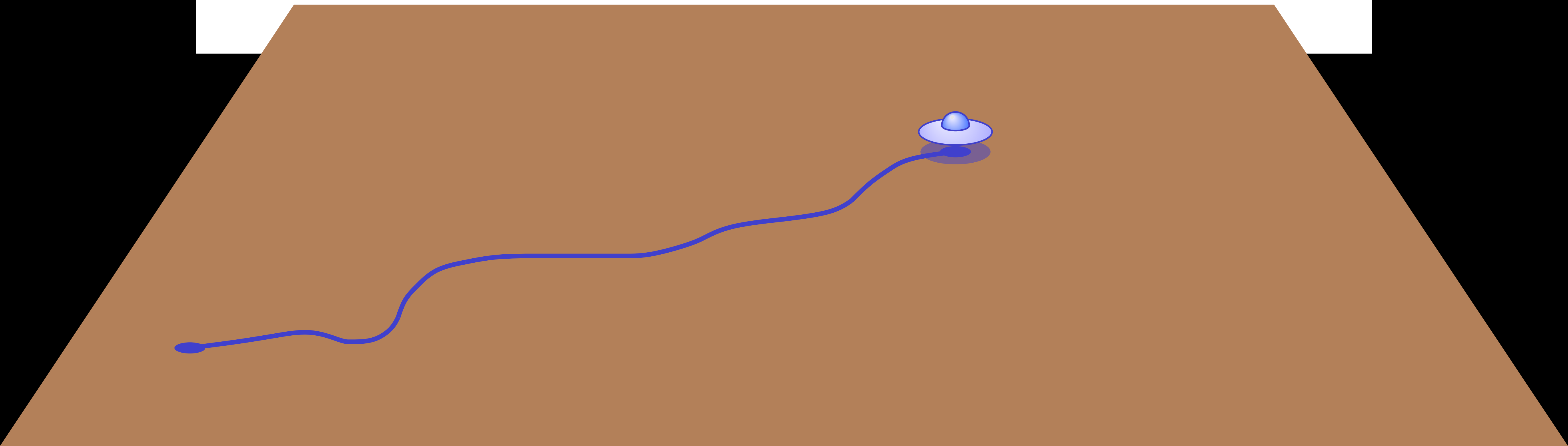
INGREDIENTS

- 1 Plane
- 1 Unpredictable Moving Object
  - Travels unpredictably in the plane



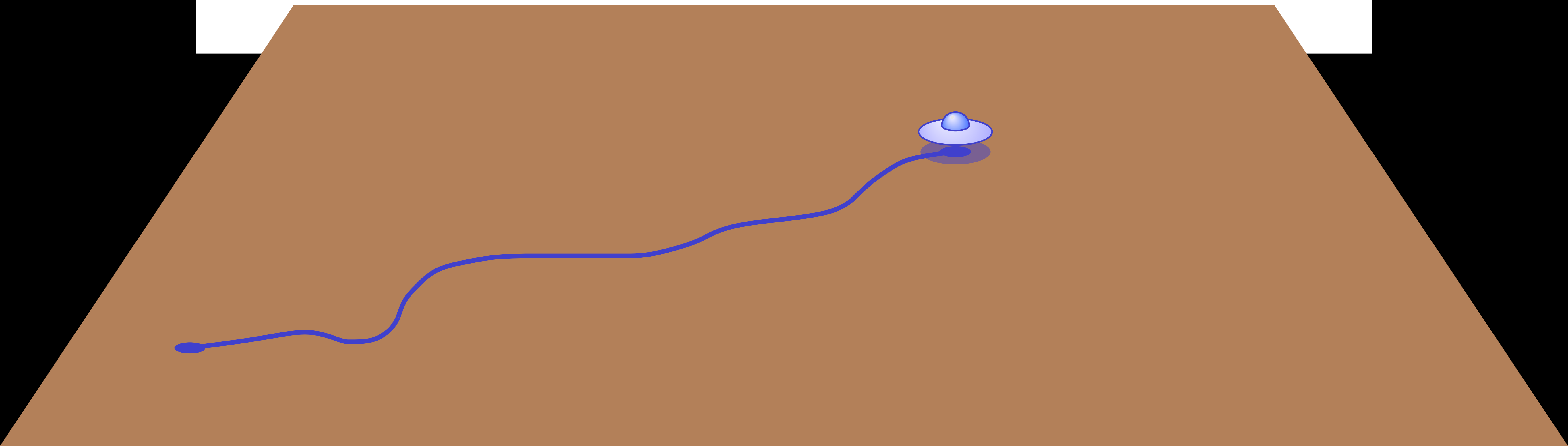
INGREDIENTS

- 1 Plane
- 1 Unpredictable Moving Object
  - Travels unpredictably in the plane



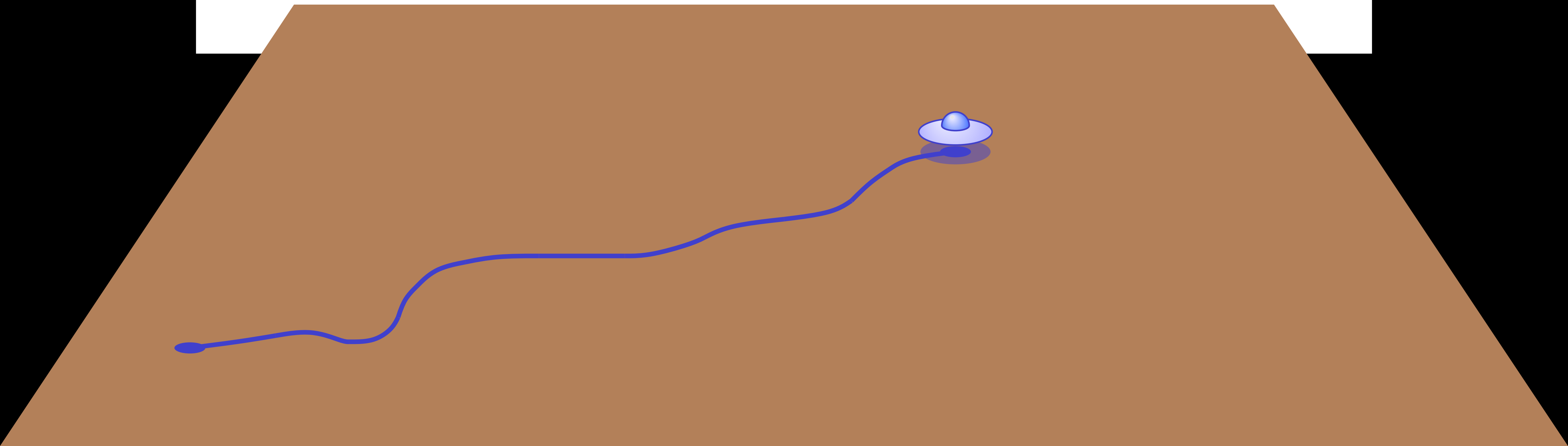
INGREDIENTS

- 1 Plane
- 1 Unpredictable Moving Object
  - Travels unpredictably in the plane
  - We wish to track the UMO



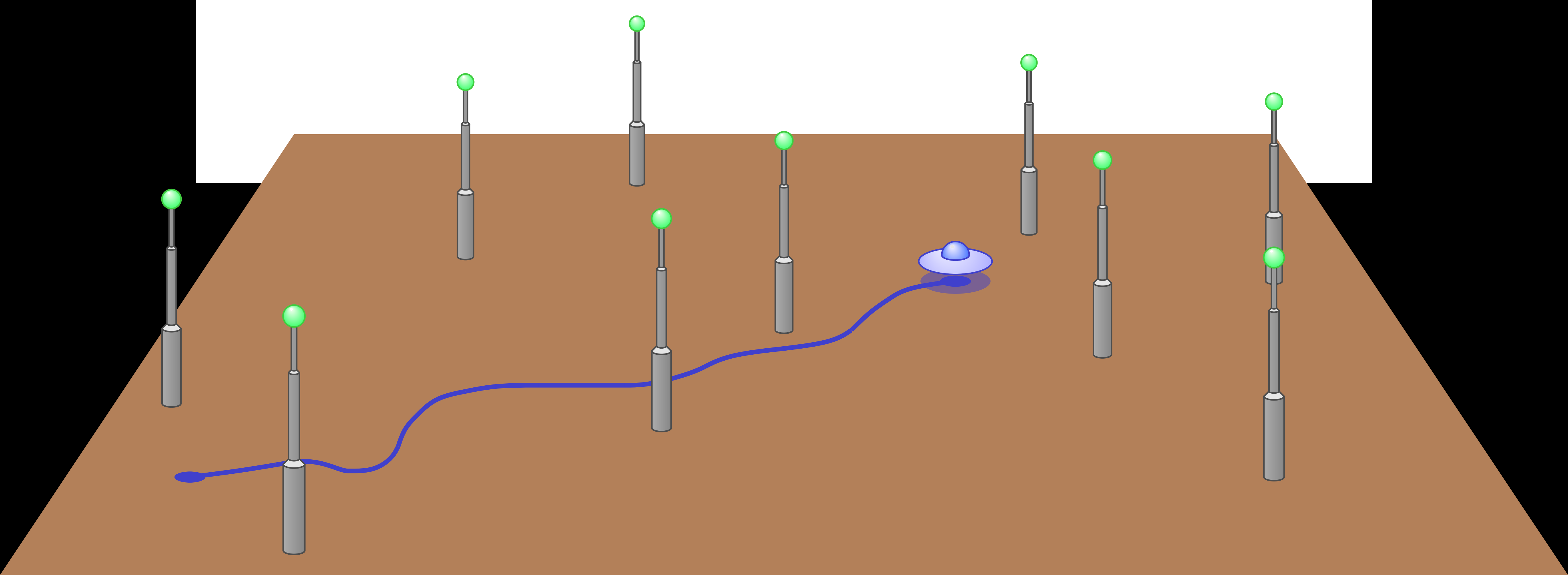
INGREDIENTS

- 1 Plane
- 1 Unpredictable Moving Object
  - Travels unpredictably in the plane
  - We wish to track the UMO
- n Sensors (Base Stations)



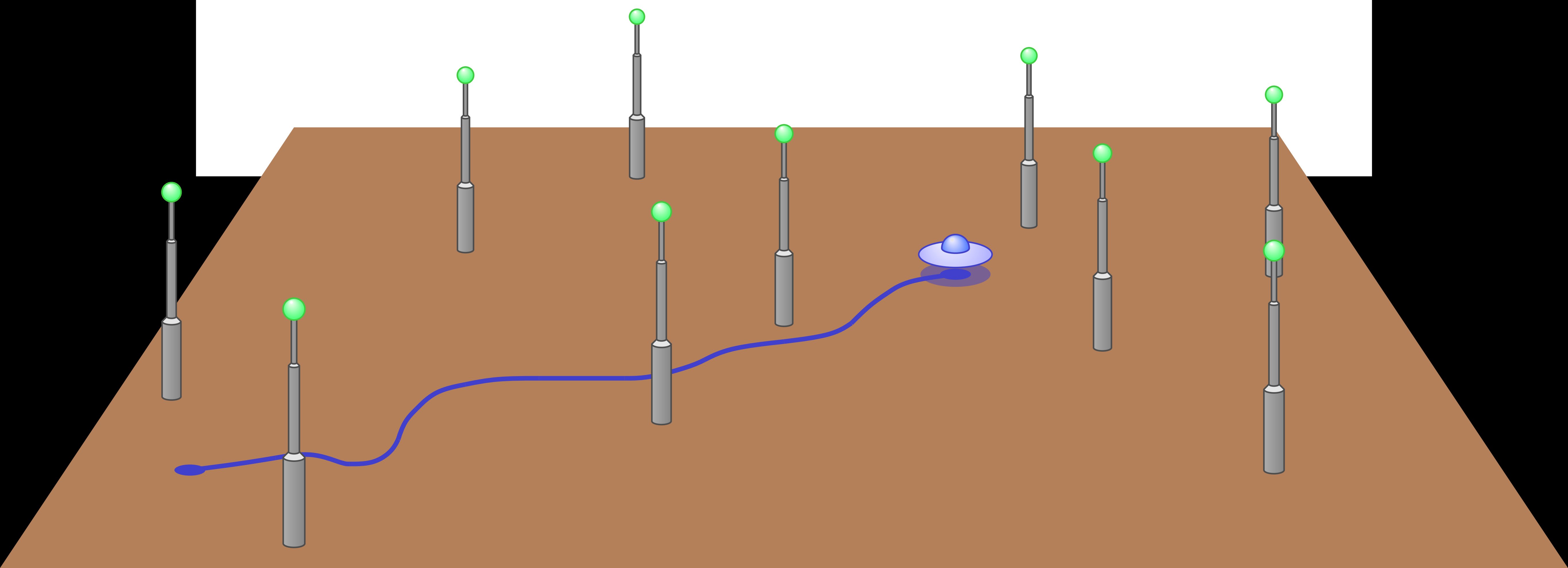
INGREDIENTS

- 1 Plane
- 1 Unpredictable Moving Object
  - Travels unpredictably in the plane
  - We wish to track the UMO
- n Sensors (Base Stations)



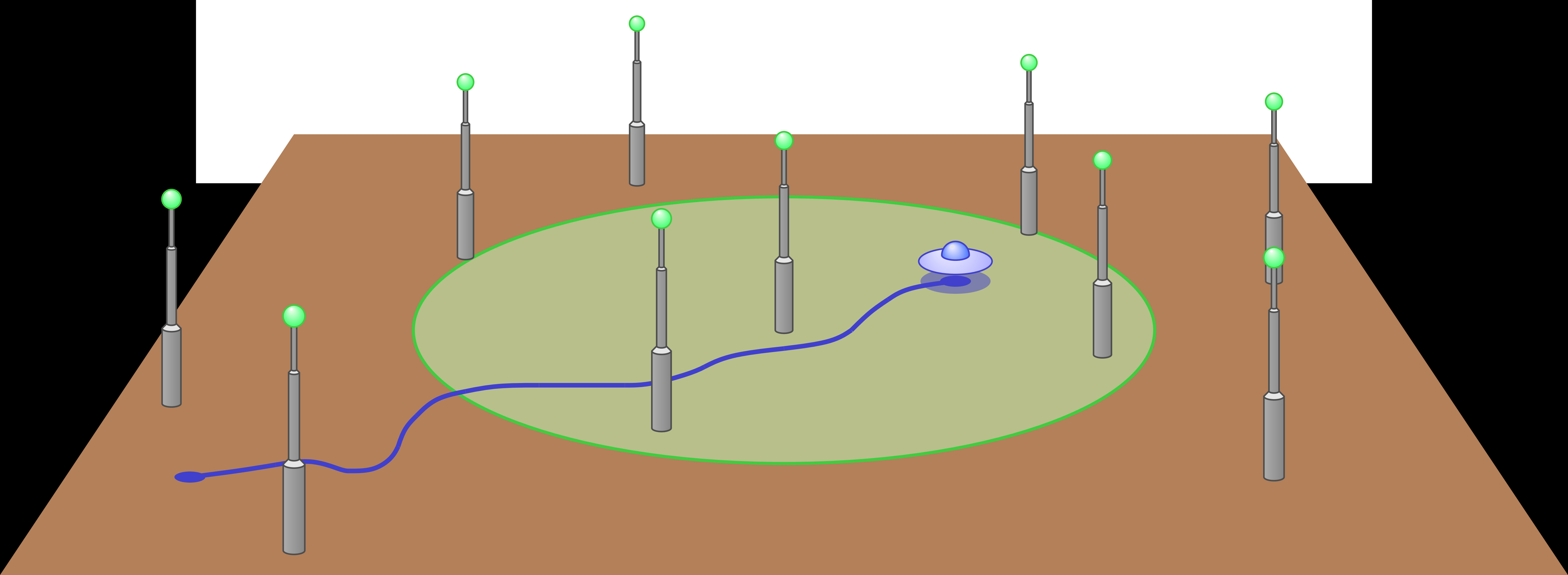
INGREDIENTS

- 1 Plane
- 1 Unpredictable Moving Object
  - Travels unpredictably in the plane
  - We wish to track the UMO
- n Sensors (Base Stations)
  - Have an associated *coverage region*



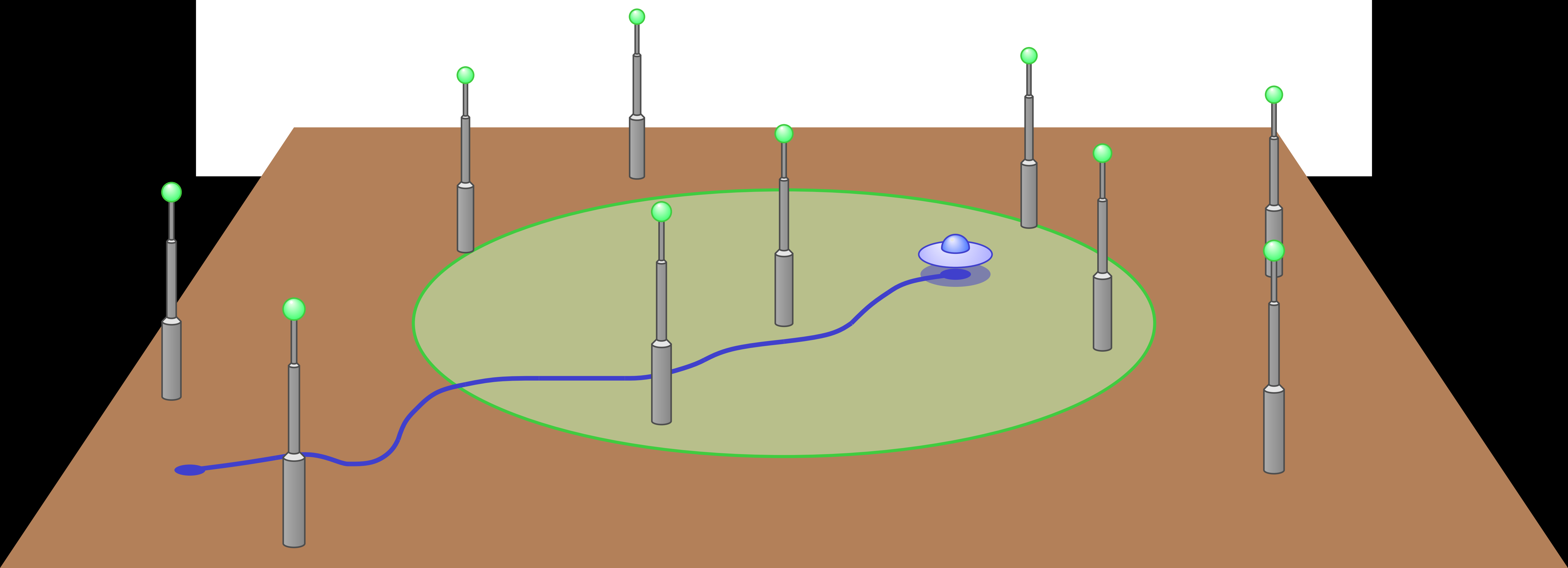
INGREDIENTS

- 1 Plane
- 1 Unpredictable Moving Object
  - Travels unpredictably in the plane
  - We wish to track the UMO
- n Sensors (Base Stations)
  - Have an associated *coverage region*



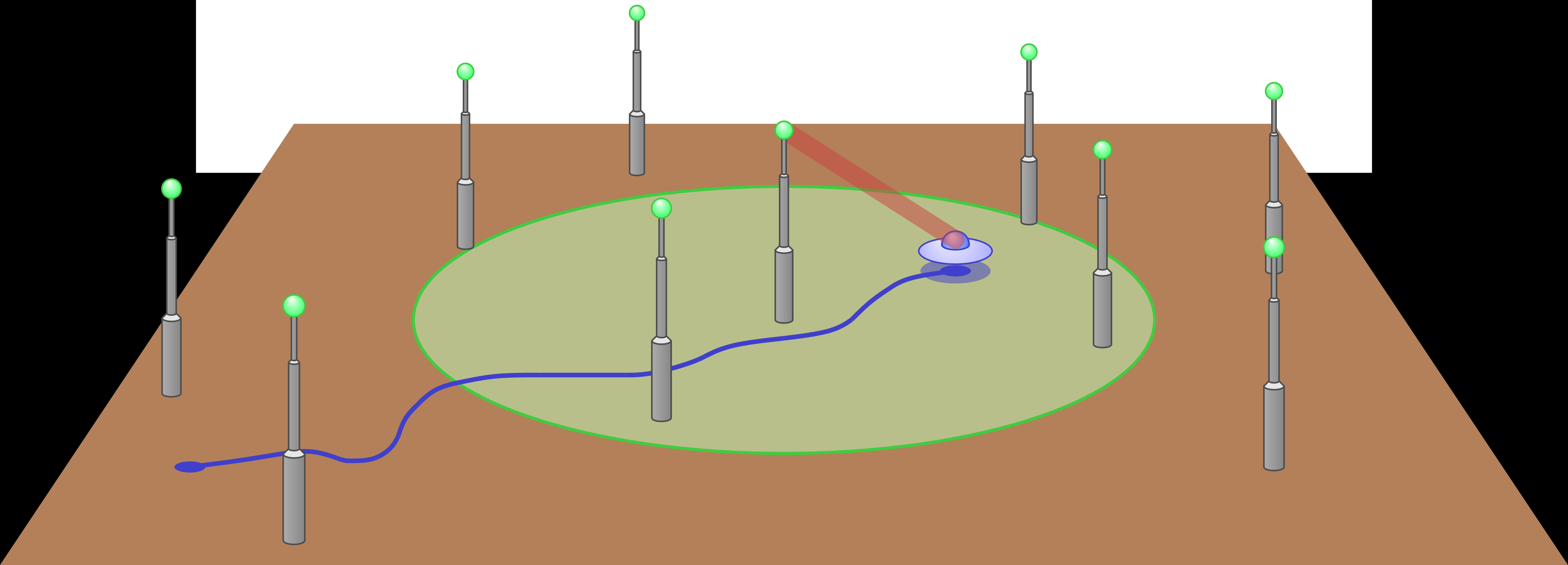
INGREDIENTS

- 1 Plane
- 1 Unpredictable Moving Object
  - Travels unpredictably in the plane
  - We wish to track the UMO
- n Sensors (Base Stations)
  - Have an associated *coverage region*
  - Can track UMO when inside region

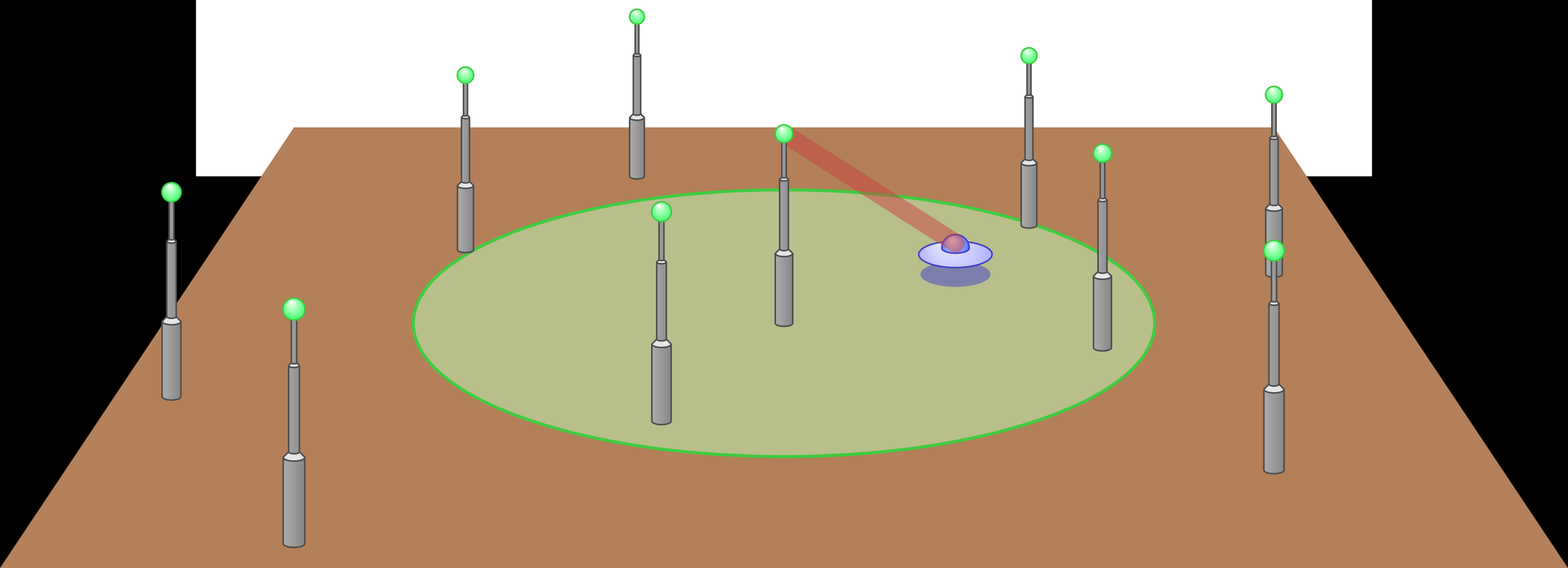


INGREDIENTS

- 1 Plane
- 1 Unpredictable Moving Object
  - Travels unpredictably in the plane
  - We wish to track the UMO
- n Sensors (Base Stations)
  - Have an associated *coverage region*
  - Can track UMO when inside region

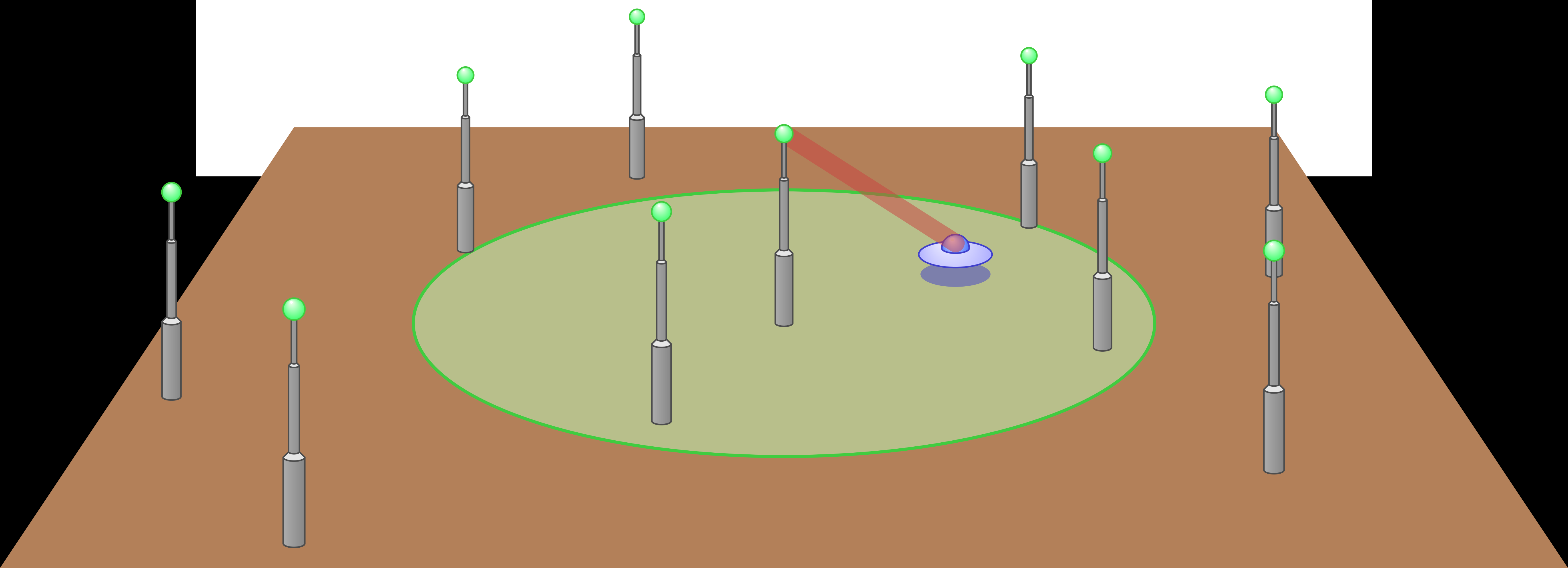


OBJECTIVES



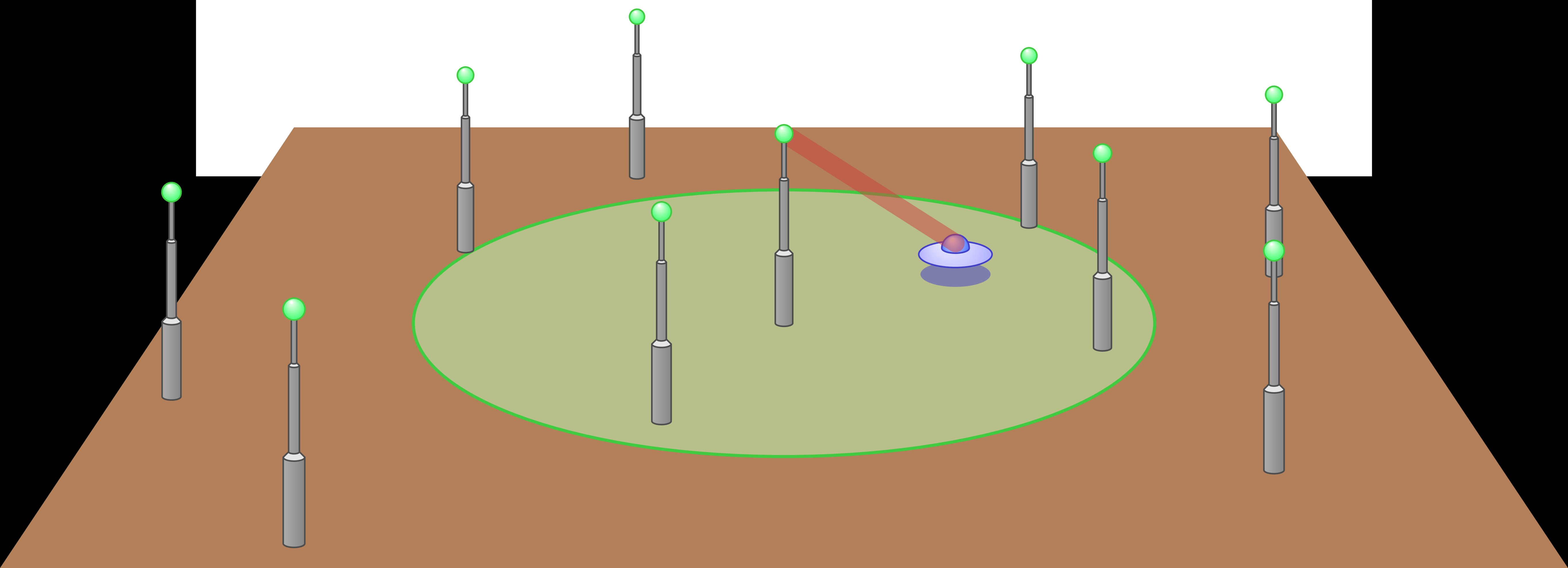
OBJECTIVES

- Trilateration



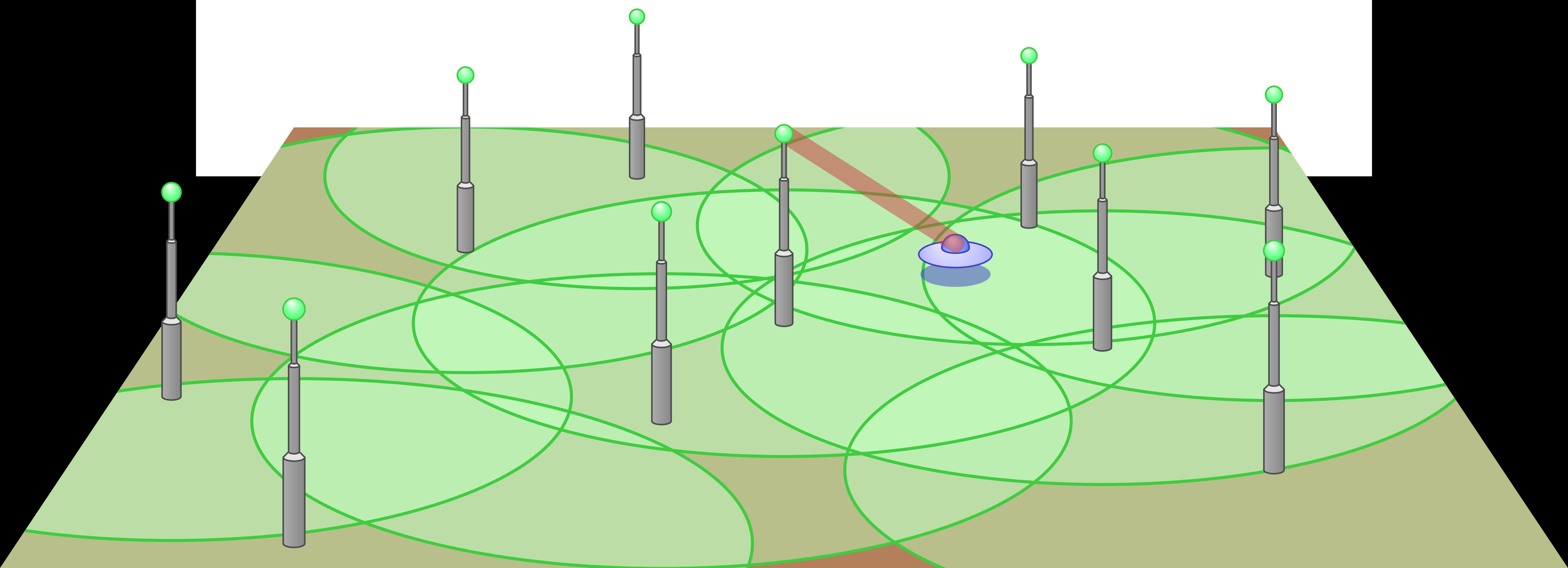
OBJECTIVES

- Trilateration
  - UMO must be tracked by 3 stations at every point in time



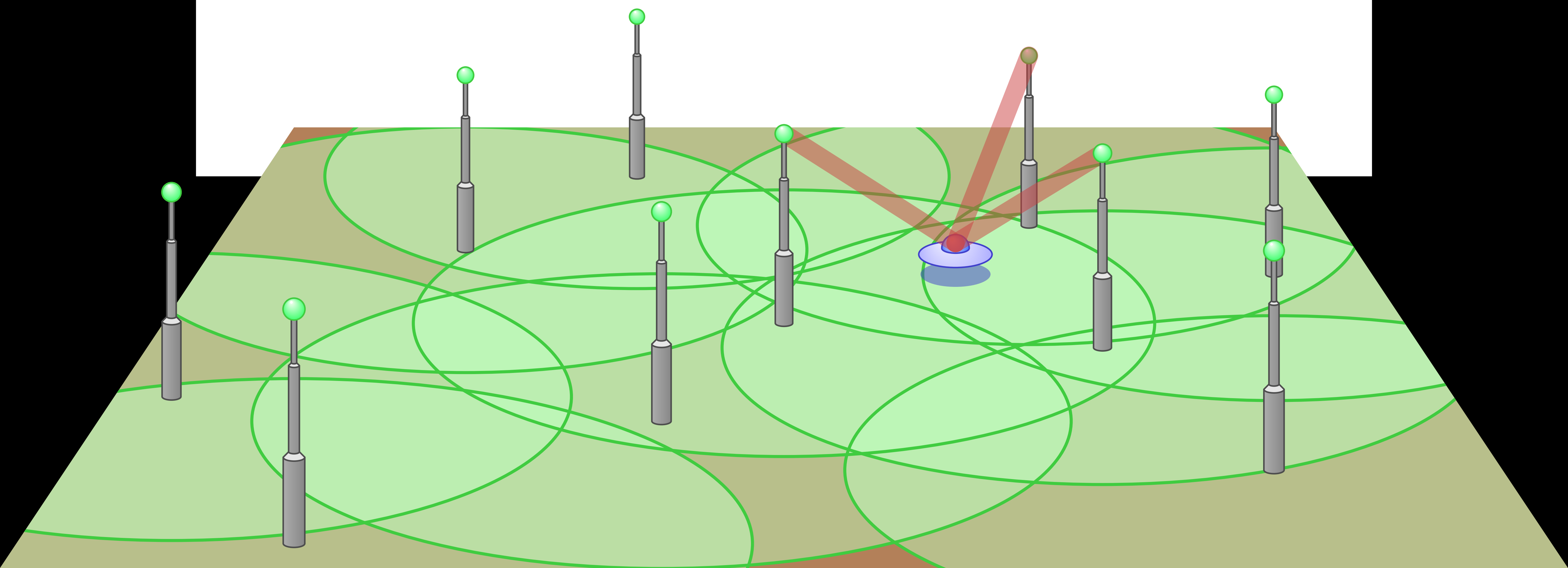
OBJECTIVES

- Trilateration
  - UMO must be tracked by 3 stations at every point in time



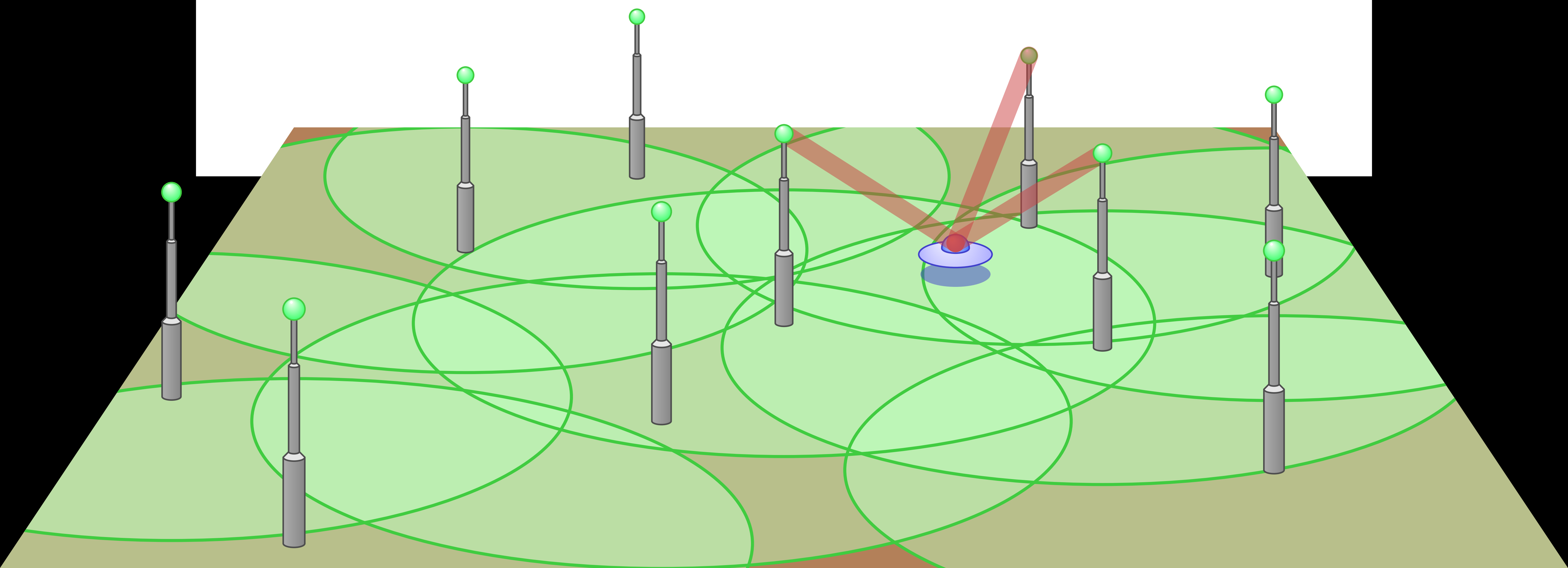
OBJECTIVES

- Trilateration
  - UMO must be tracked by 3 stations at every point in time



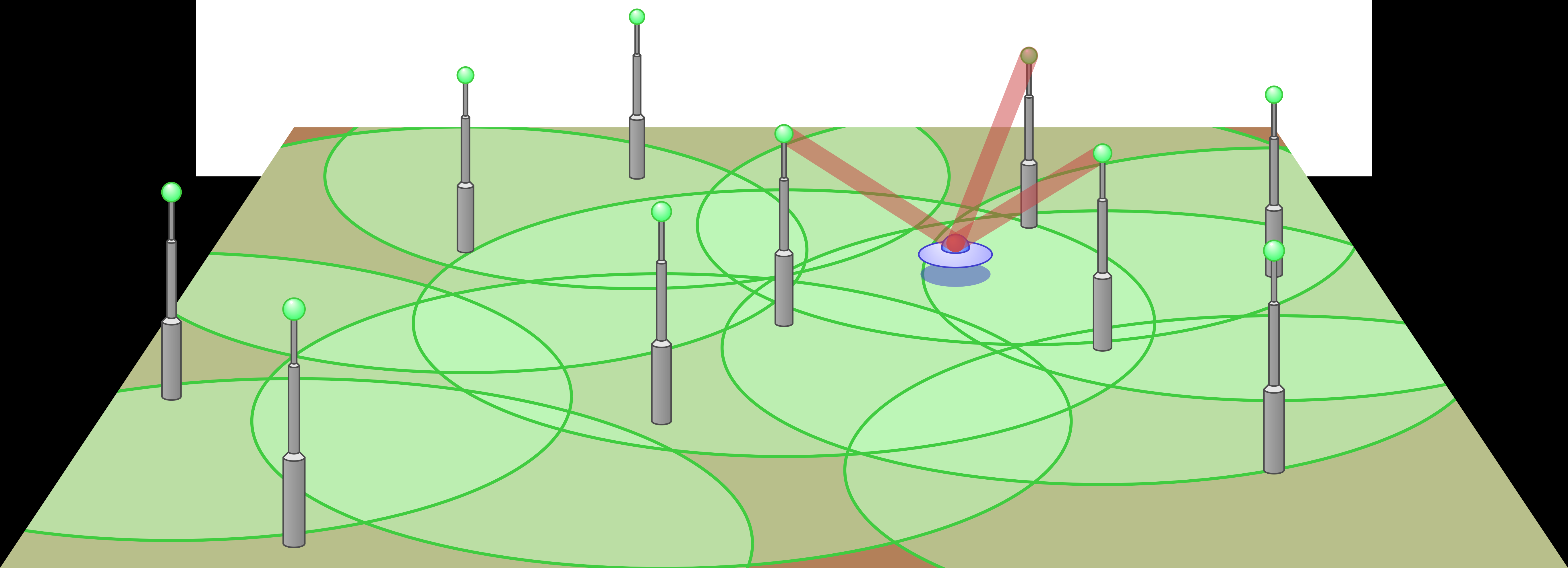
OBJECTIVES

- Trilateration
  - UMO must be tracked by 3 stations at every point in time [Yang & Liu, 2010]



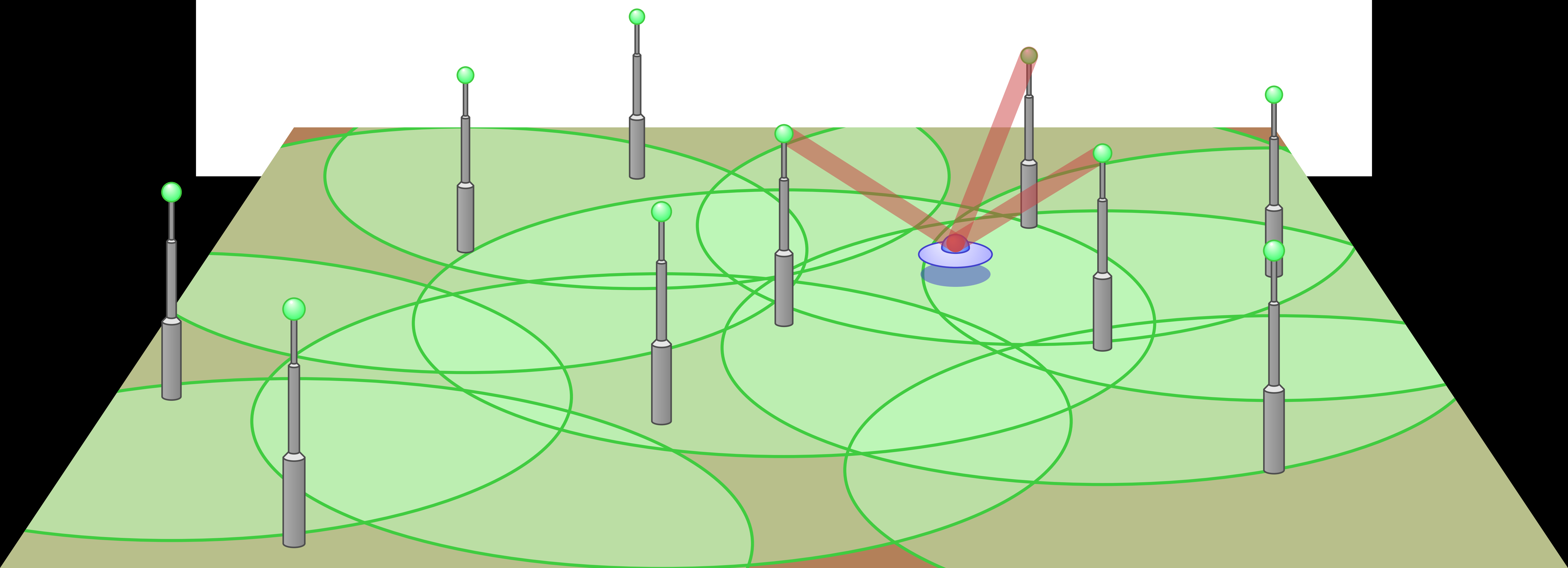
OBJECTIVES

- ~~Trilateration~~ c-lateration
  - UMO must be tracked by ~~3~~ c stations at every point in time



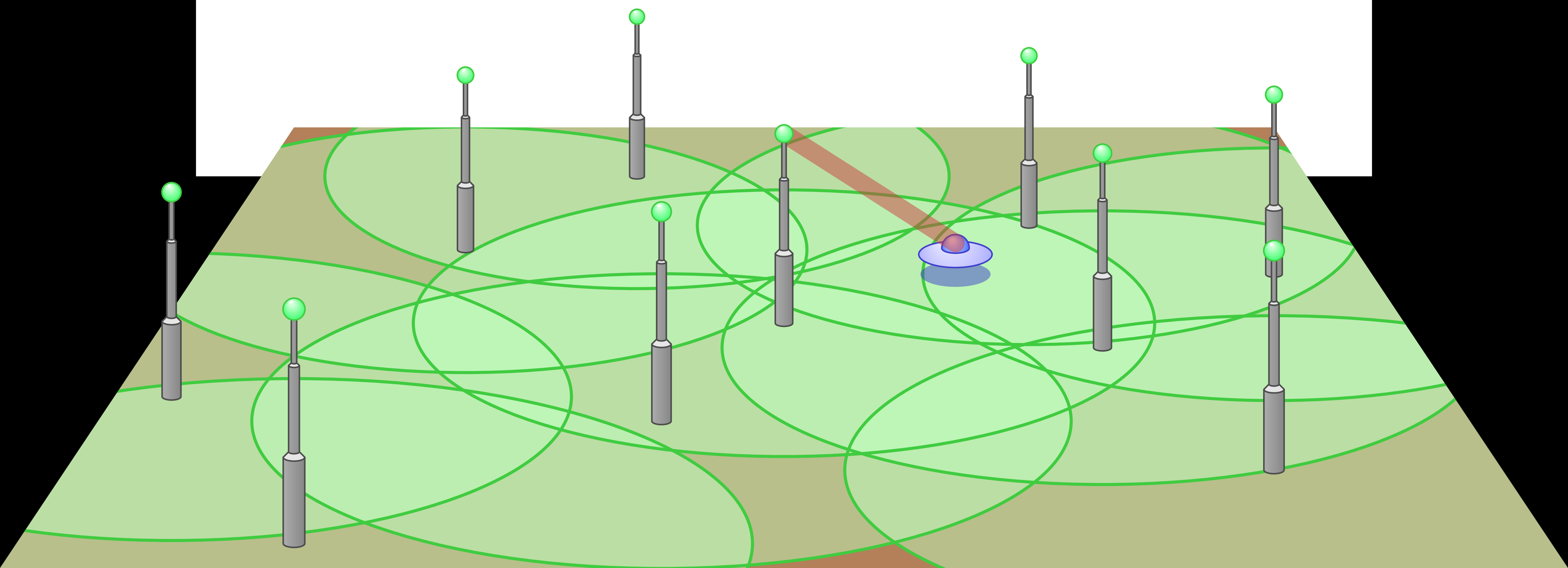
OBJECTIVES

- ~~Trilateration~~ c-lateration
  - UMO must be tracked by ~~3~~ c stations at every point in time
  - Let's focus on $c = 1$



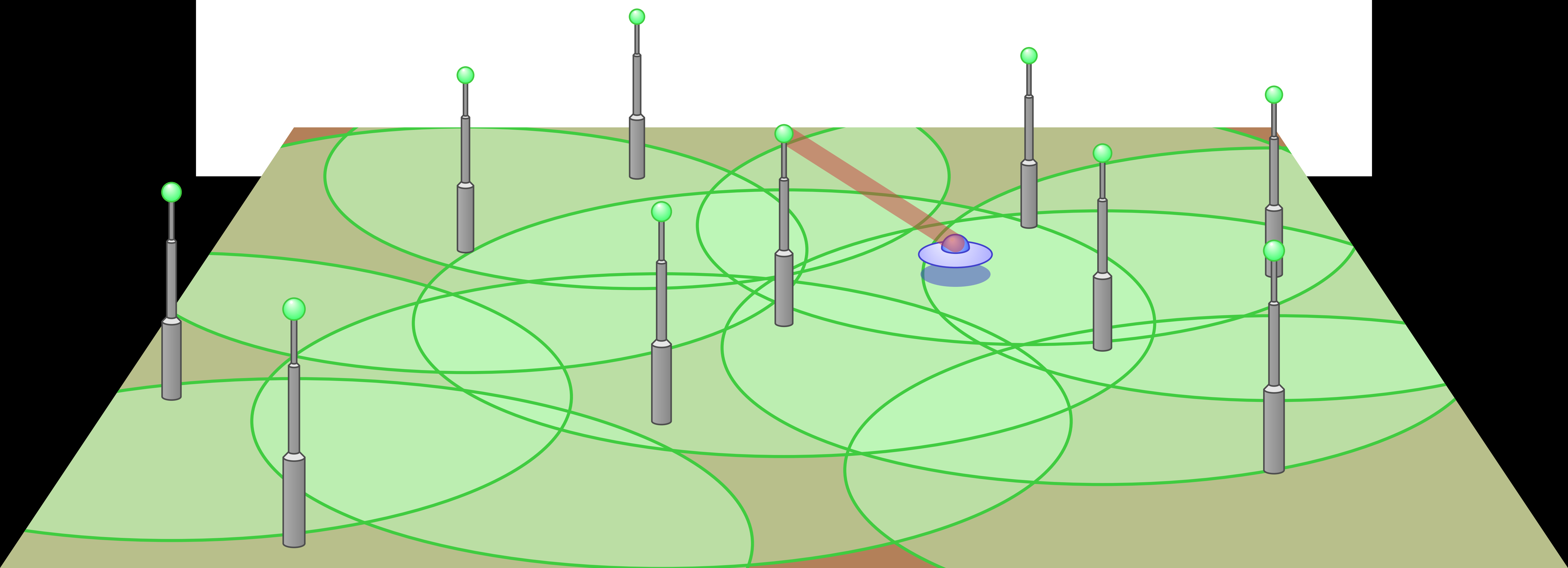
OBJECTIVES

- ~~Trilateration~~ c-lateration
  - UMO must be tracked by ~~3~~ c stations at every point in time
  - Let's focus on $c = 1$



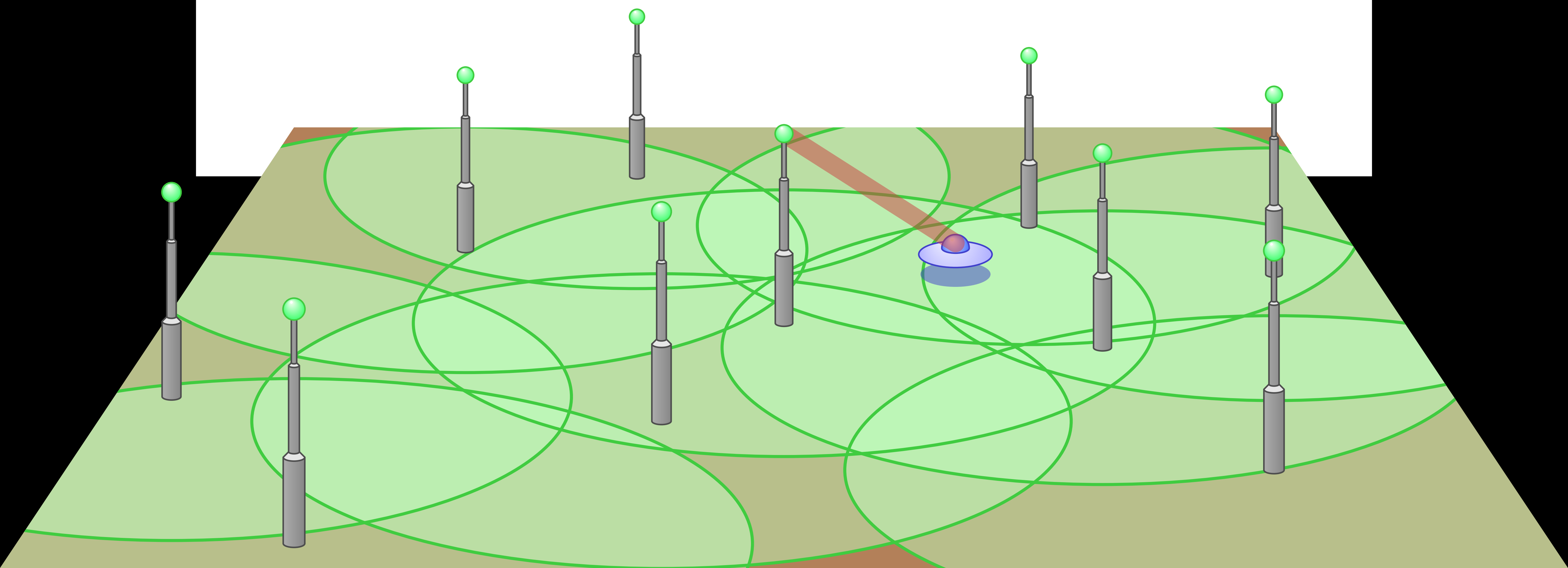
OBJECTIVES

- ~~Trilateration~~ c-lateration
  - UMO must be tracked by ~~3~~ c stations at every point in time
  - Let's focus on $c = 1$
- Handovers



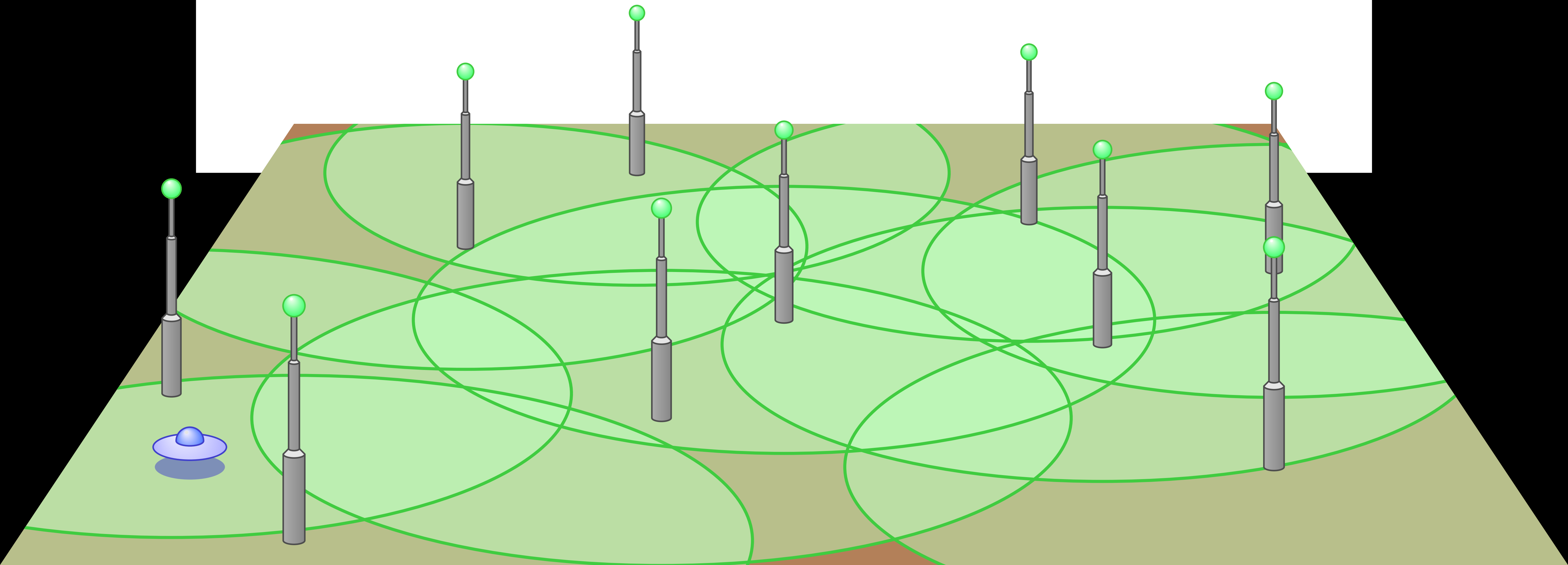
OBJECTIVES

- ~~Trilateration~~ c-lateration
  - UMO must be tracked by ~~3~~ c stations at every point in time
  - Let's focus on $c = 1$
- Handovers
  - When UMO leaves sensor region, we need to assign it to a different sensor



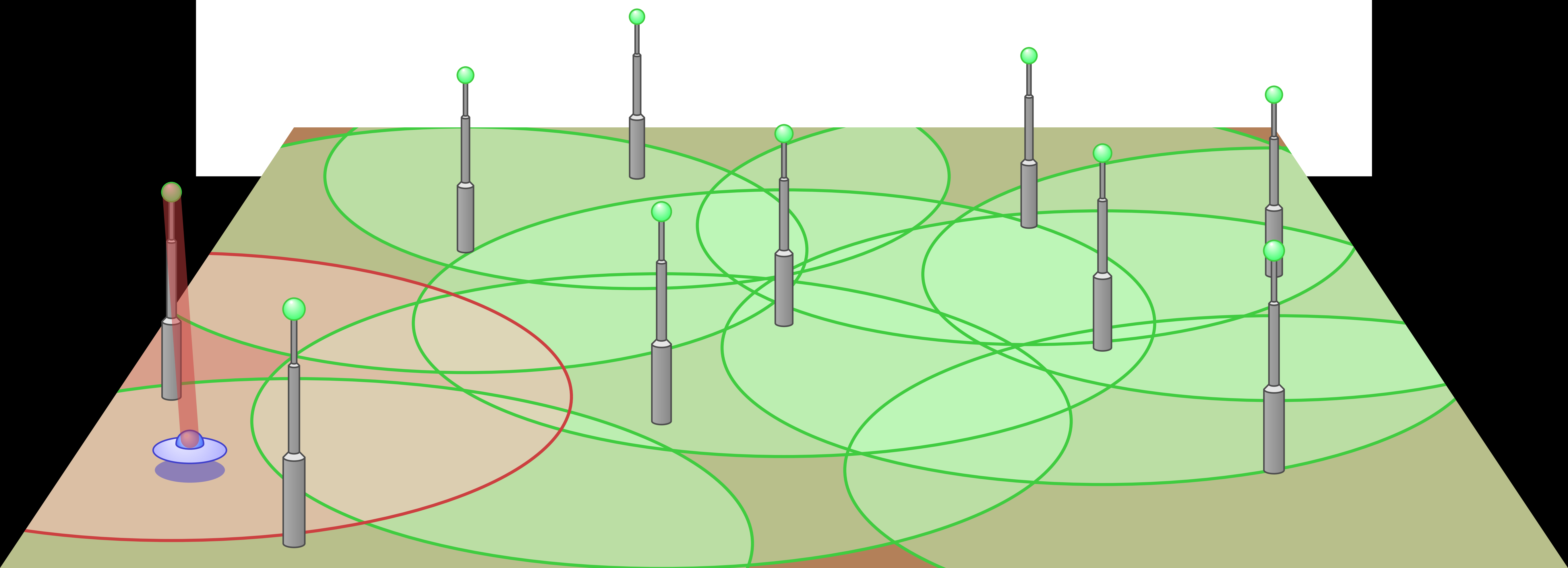
OBJECTIVES

- ~~Trilateration~~ c-lateration
  - UMO must be tracked by ~~3~~ c stations at every point in time
  - Let's focus on $c = 1$
- Handovers
  - When UMO leaves sensor region, we need to assign it to a different sensor



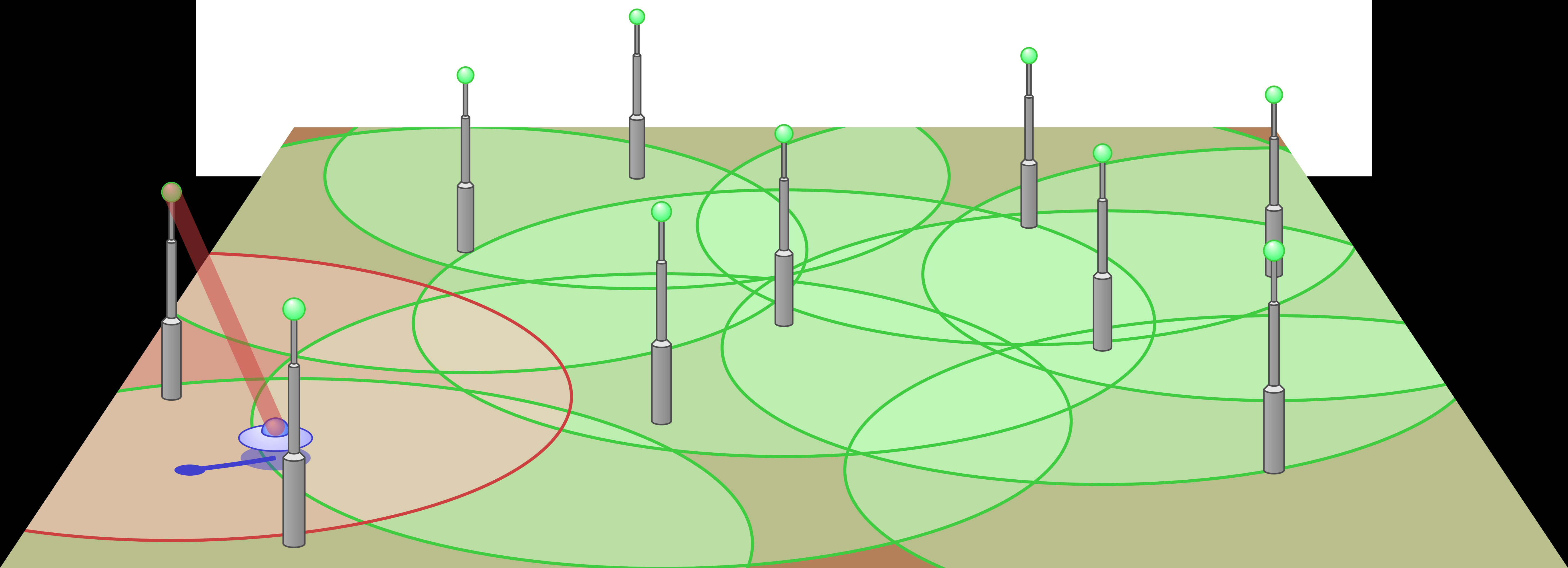
OBJECTIVES

- ~~Trilateration~~ c-lateration
  - UMO must be tracked by ~~3~~ c stations at every point in time
  - Let's focus on $c = 1$
- Handovers
  - When UMO leaves sensor region, we need to assign it to a different sensor



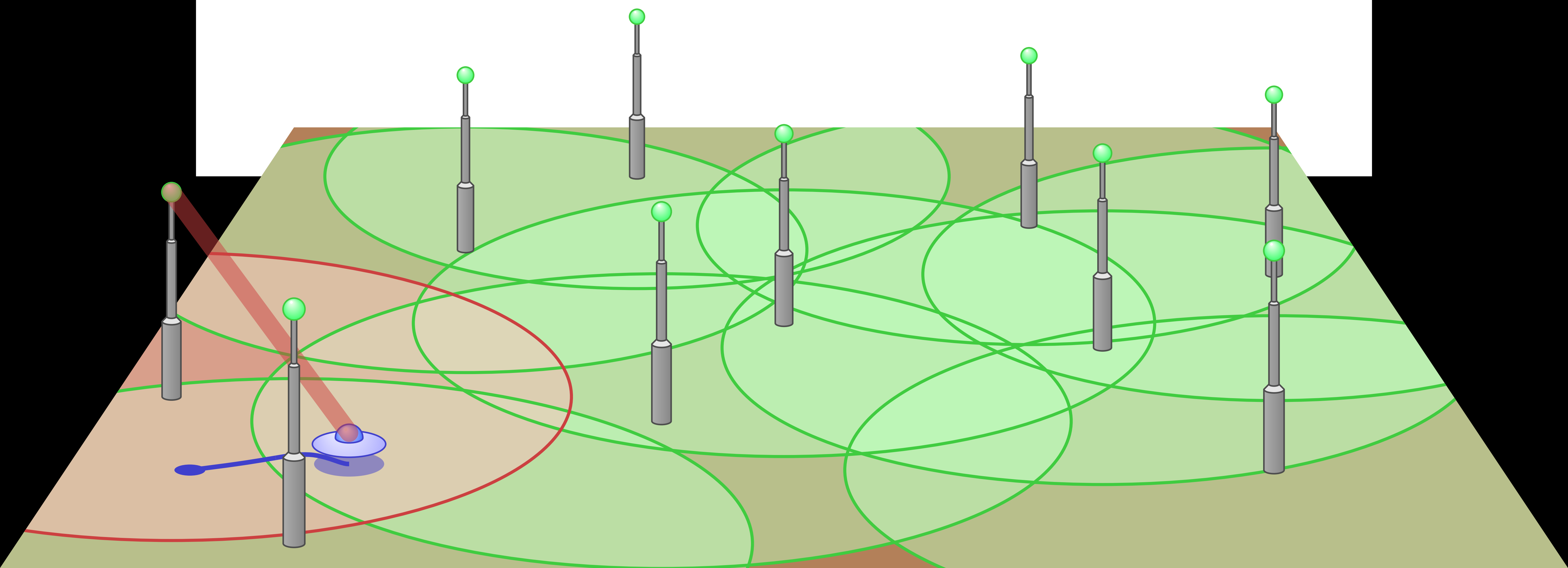
OBJECTIVES

- ~~Trilateration~~ c-lateration
  - UMO must be tracked by ~~3~~ c stations at every point in time
  - Let's focus on $c = 1$
- Handovers
  - When UMO leaves sensor region, we need to assign it to a different sensor



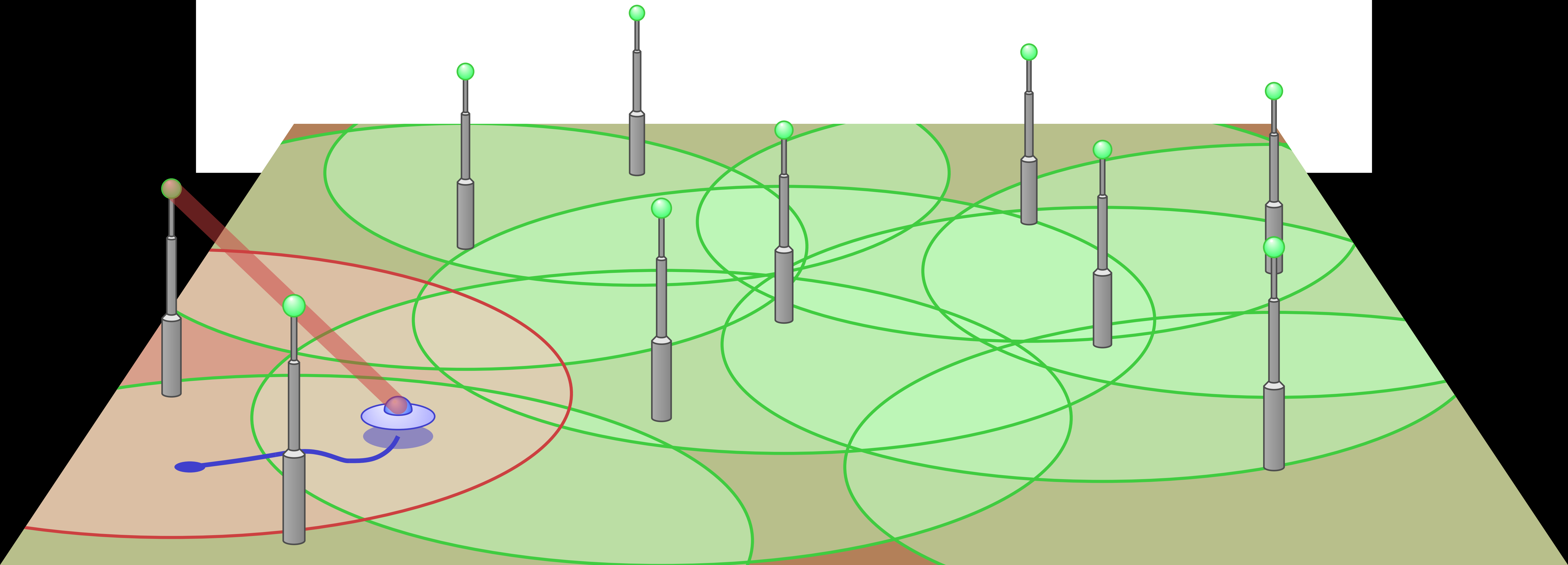
OBJECTIVES

- ~~Trilateration~~ c-lateration
  - UMO must be tracked by ~~3~~ c stations at every point in time
  - Let's focus on $c = 1$
- Handovers
  - When UMO leaves sensor region, we need to assign it to a different sensor



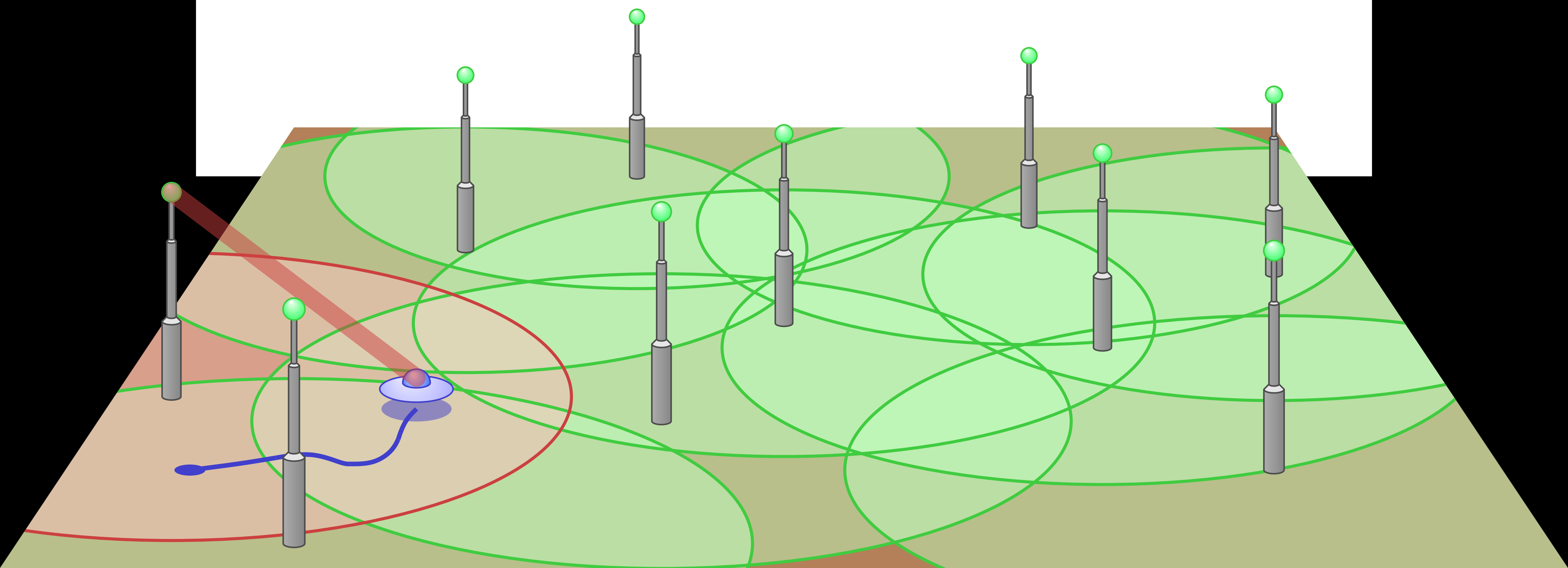
OBJECTIVES

- ~~Trilateration~~ c-lateration
  - UMO must be tracked by ~~3~~ c stations at every point in time
  - Let's focus on $c = 1$
- Handovers
  - When UMO leaves sensor region, we need to assign it to a different sensor



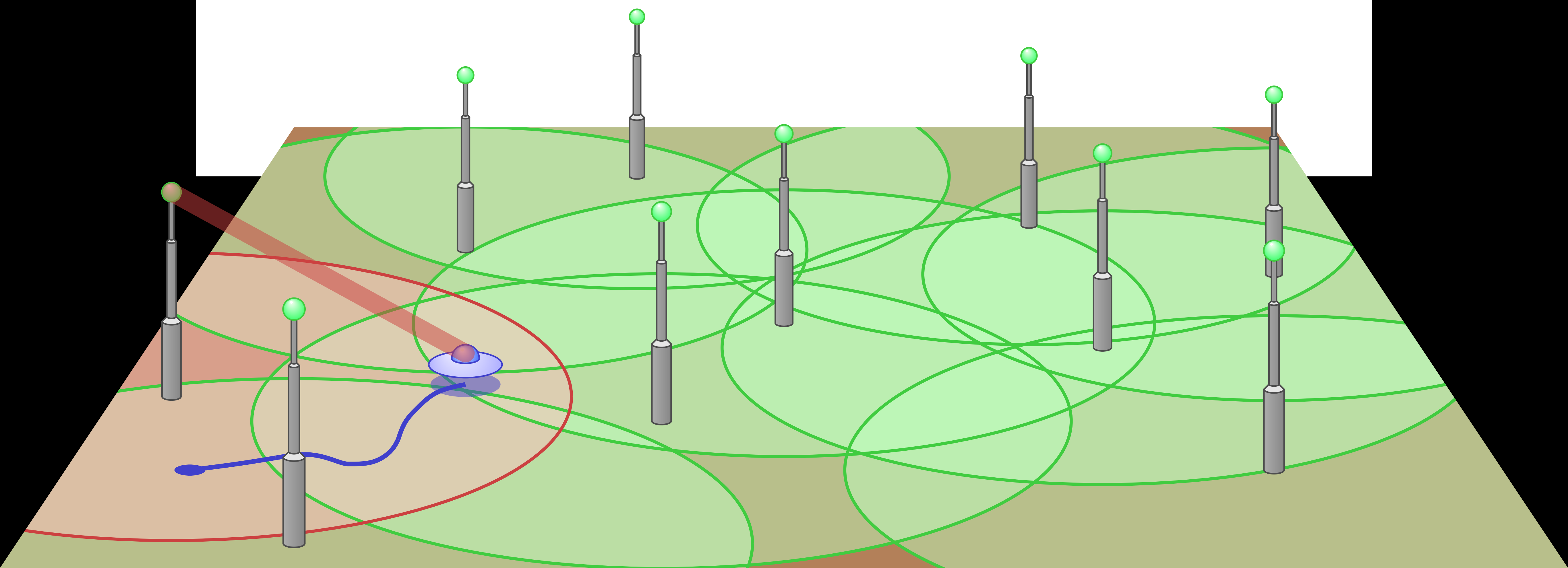
OBJECTIVES

- ~~Trilateration~~ c-lateration
  - UMO must be tracked by ~~3~~ c stations at every point in time
  - Let's focus on $c = 1$
- Handovers
  - When UMO leaves sensor region, we need to assign it to a different sensor



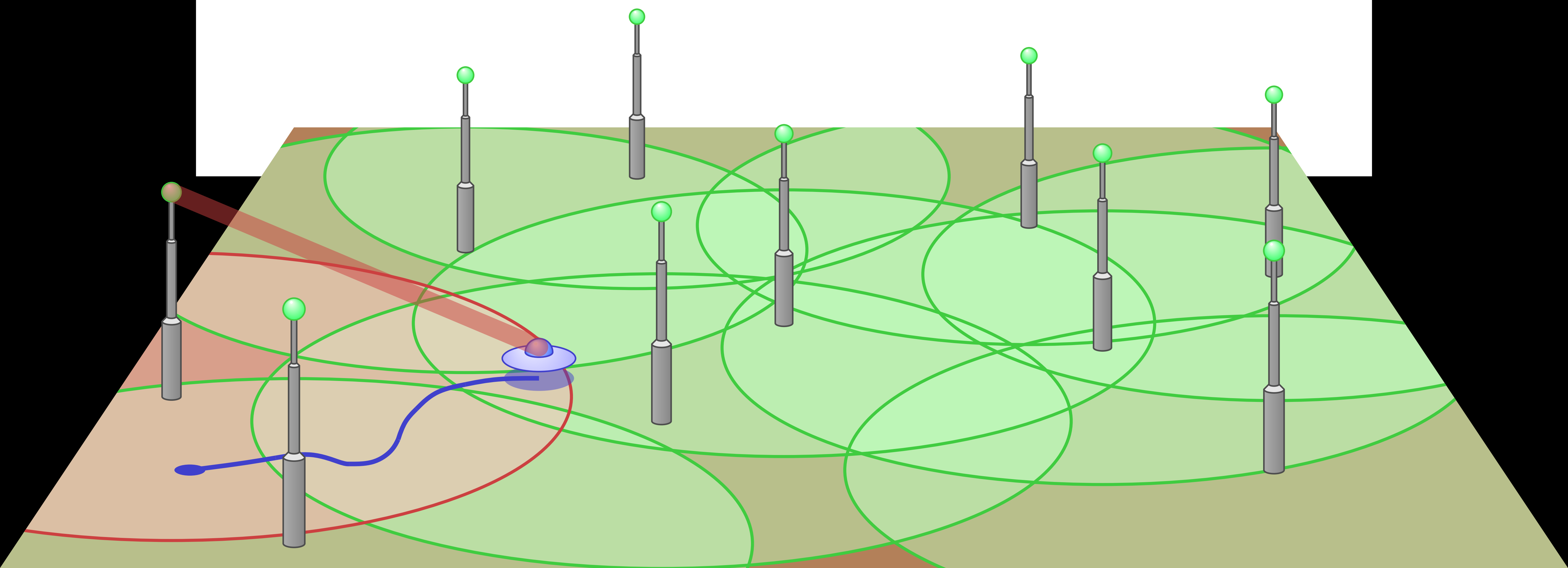
OBJECTIVES

- ~~Trilateration~~ c-lateration
  - UMO must be tracked by ~~3~~ c stations at every point in time
  - Let's focus on $c = 1$
- Handovers
  - When UMO leaves sensor region, we need to assign it to a different sensor



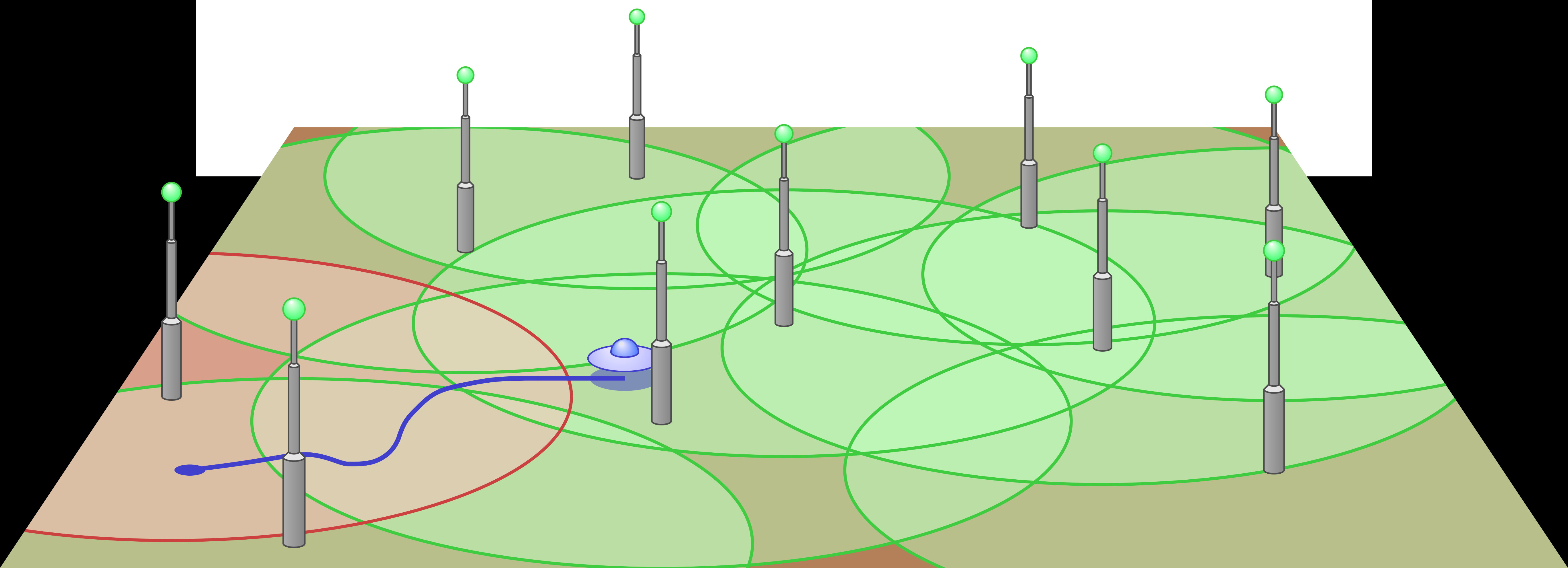
OBJECTIVES

- ~~Trilateration~~ c-lateration
  - UMO must be tracked by ~~3~~ c stations at every point in time
  - Let's focus on $c = 1$
- Handovers
  - When UMO leaves sensor region, we need to assign it to a different sensor



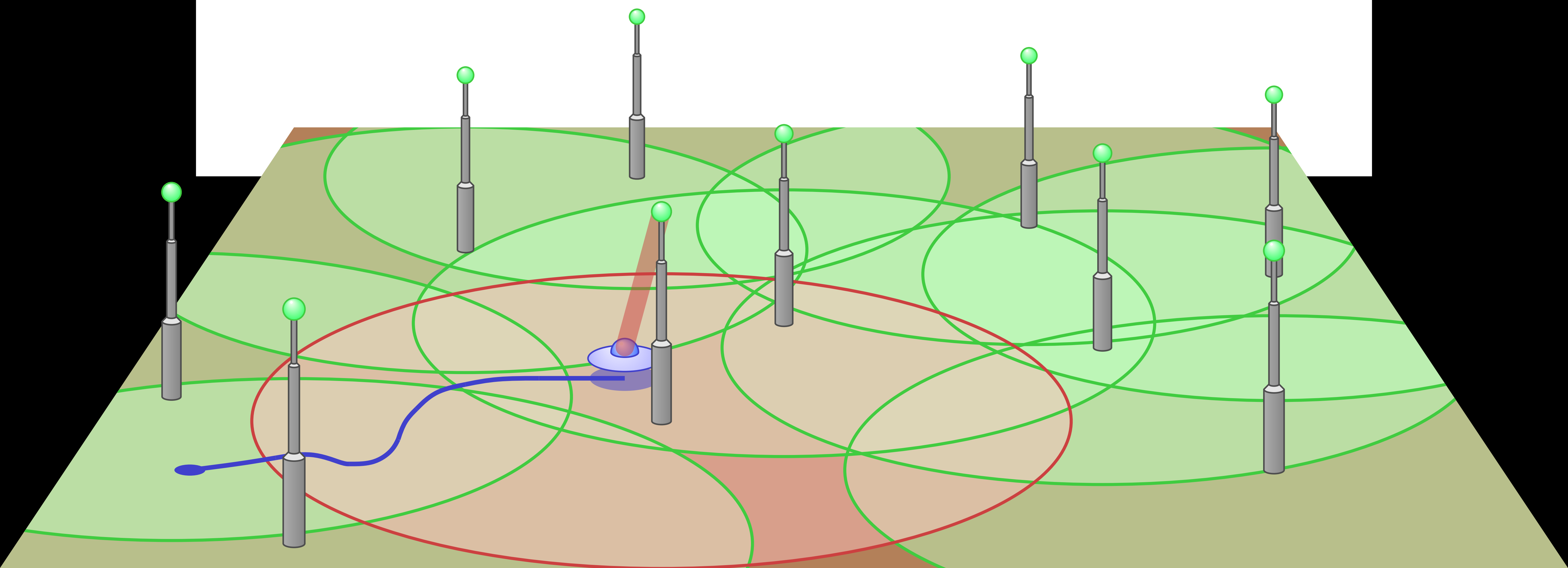
OBJECTIVES

- ~~Trilateration~~ c-lateration
  - UMO must be tracked by ~~3~~ c stations at every point in time
  - Let's focus on $c = 1$
- Handovers
  - When UMO leaves sensor region, we need to assign it to a different sensor



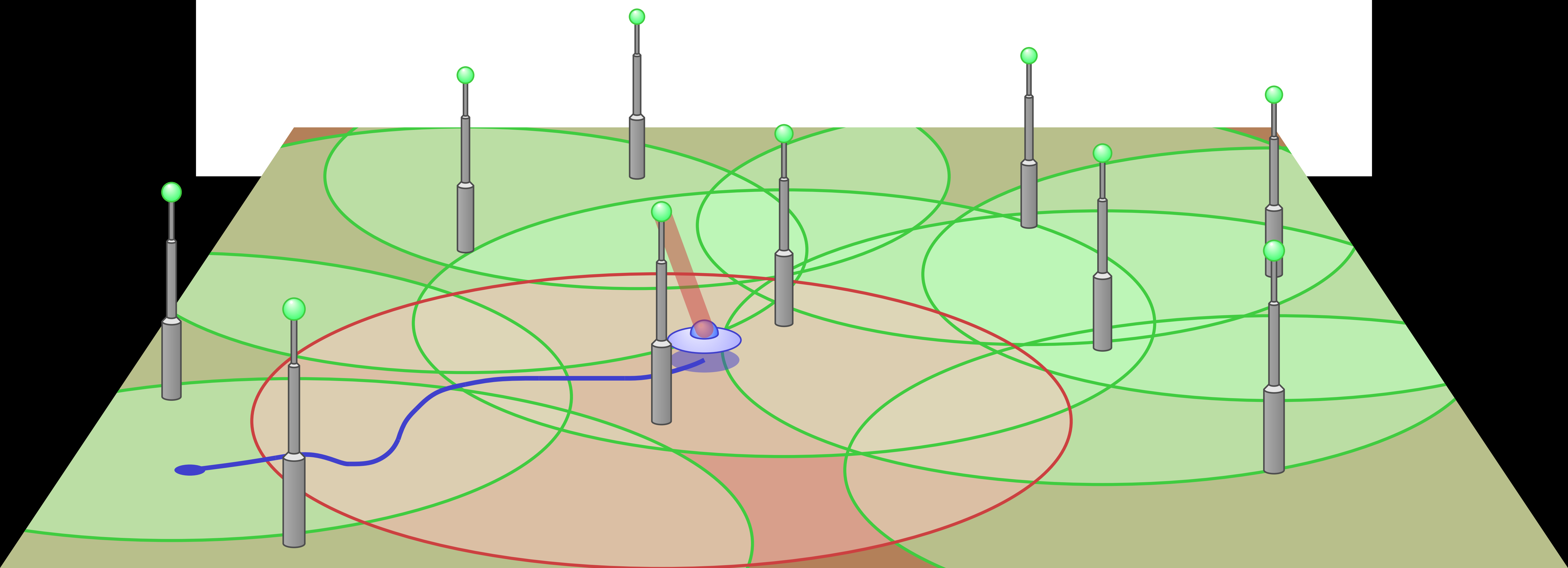
OBJECTIVES

- ~~Trilateration~~ c-lateration
  - UMO must be tracked by ~~3~~ c stations at every point in time
  - Let's focus on $c = 1$
- Handovers
  - When UMO leaves sensor region, we need to assign it to a different sensor



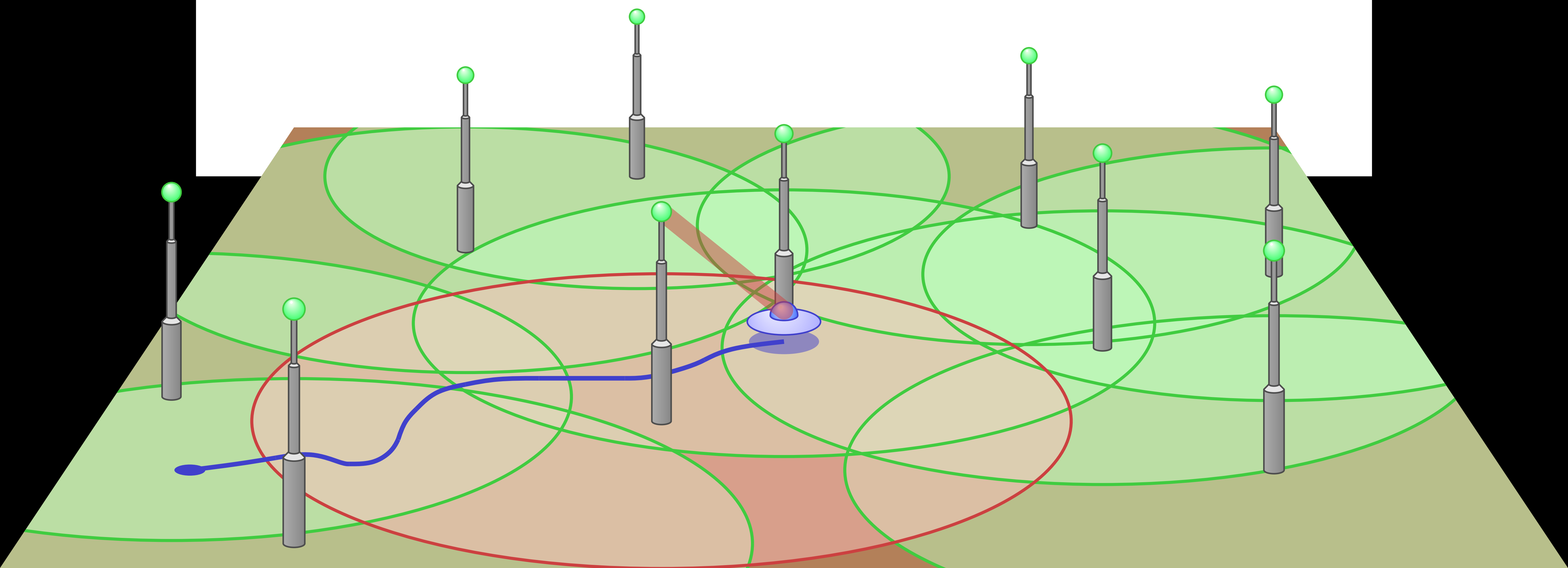
OBJECTIVES

- ~~Trilateration~~ c-lateration
  - UMO must be tracked by ~~3~~ c stations at every point in time
  - Let's focus on $c = 1$
- Handovers
  - When UMO leaves sensor region, we need to assign it to a different sensor



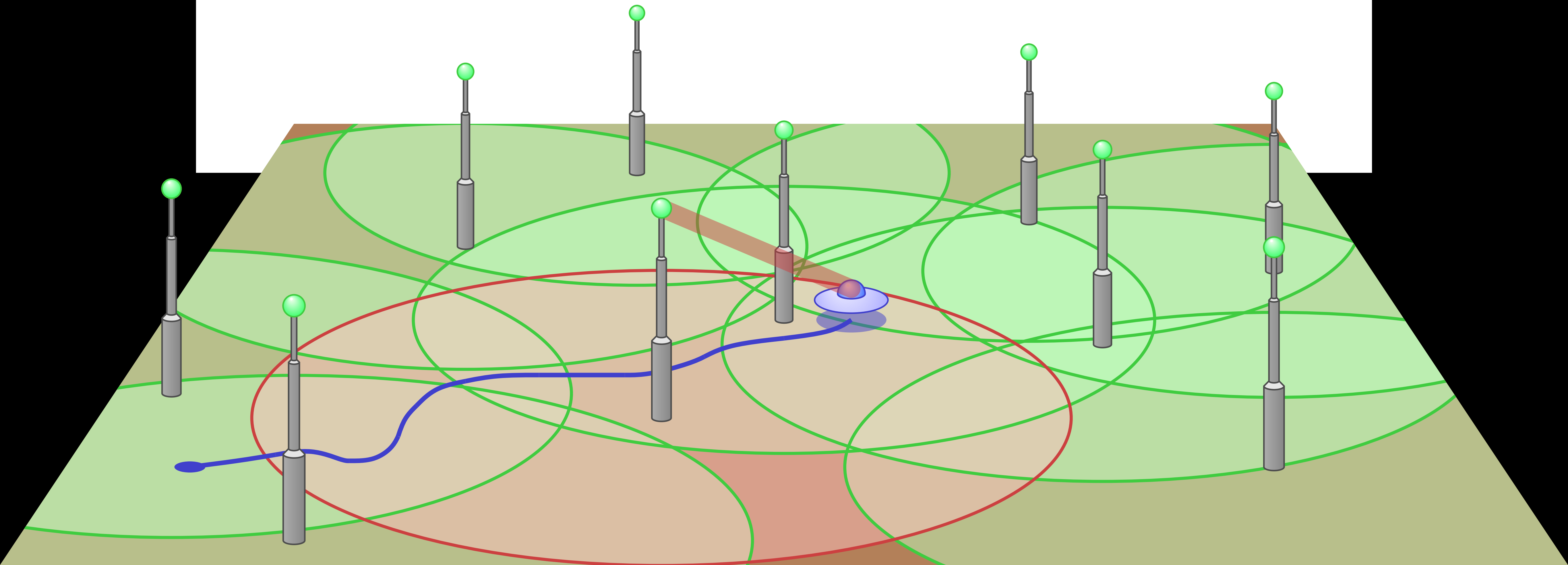
OBJECTIVES

- Trilateration c-lateration
  - UMO must be tracked by 3 c stations at every point in time
  - Let's focus on $c = 1$
- Handovers
  - When UMO leaves sensor region, we need to assign it to a different sensor



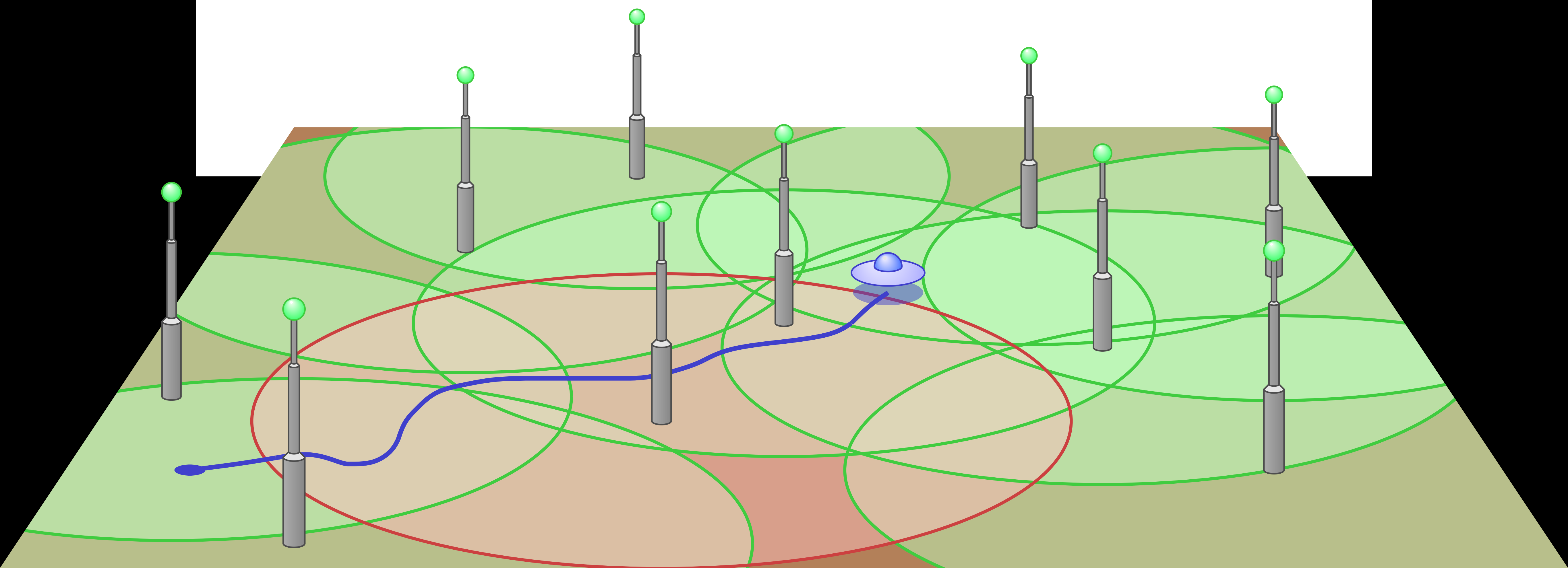
OBJECTIVES

- ~~Trilateration~~ c-lateration
  - UMO must be tracked by ~~3~~ c stations at every point in time
  - Let's focus on $c = 1$
- Handovers
  - When UMO leaves sensor region, we need to assign it to a different sensor



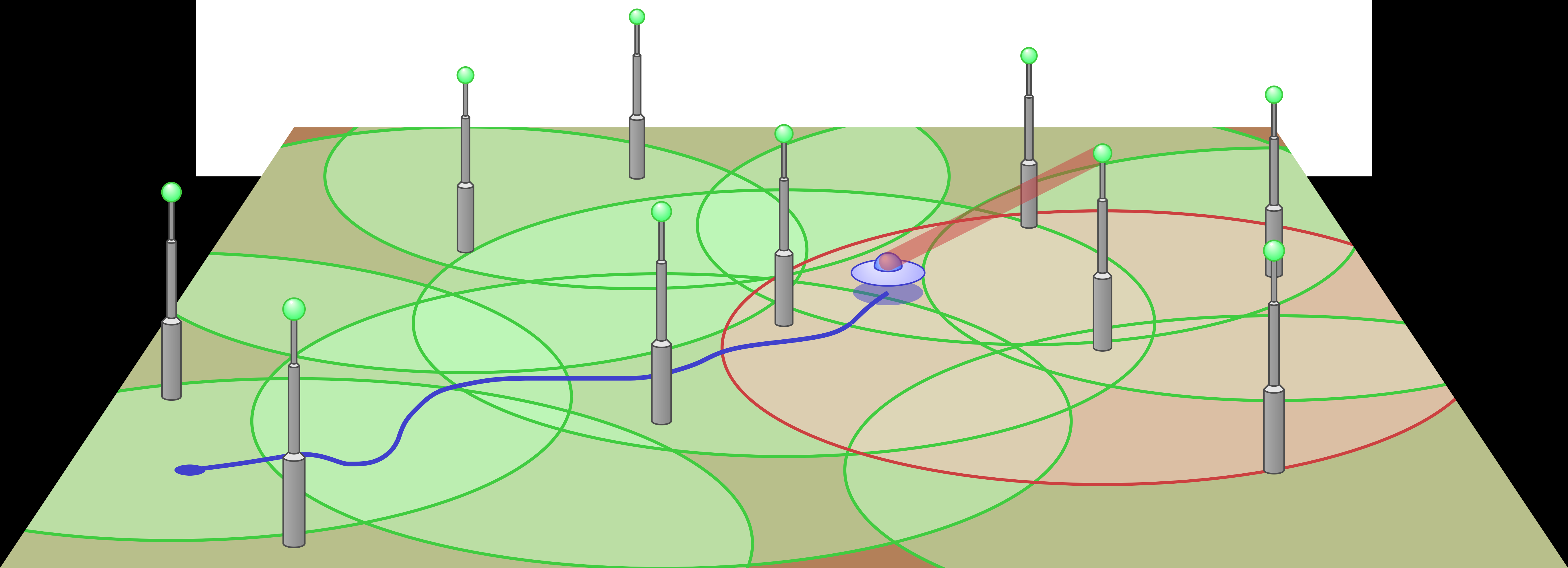
OBJECTIVES

- ~~Trilateration~~ c-lateration
  - UMO must be tracked by ~~3~~ c stations at every point in time
  - Let's focus on $c = 1$
- Handovers
  - When UMO leaves sensor region, we need to assign it to a different sensor



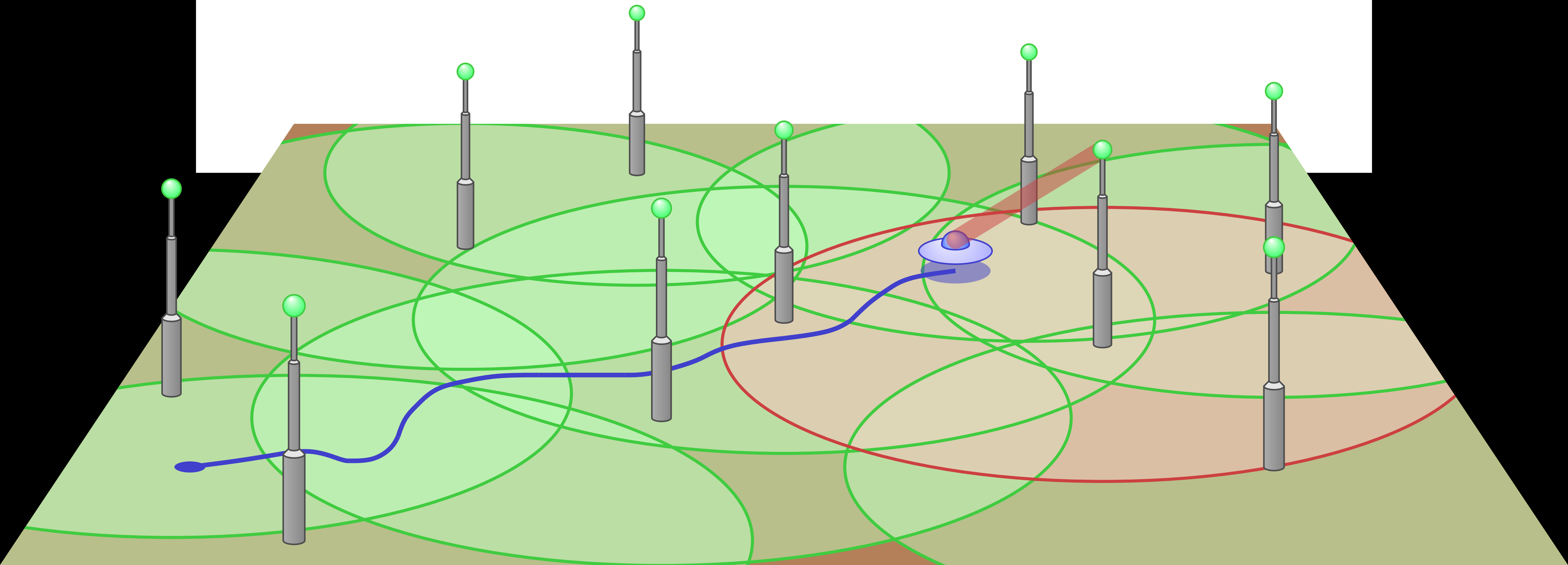
OBJECTIVES

- ~~Trilateration~~ c-lateration
  - UMO must be tracked by ~~3~~ c stations at every point in time
  - Let's focus on $c = 1$
- Handovers
  - When UMO leaves sensor region, we need to assign it to a different sensor



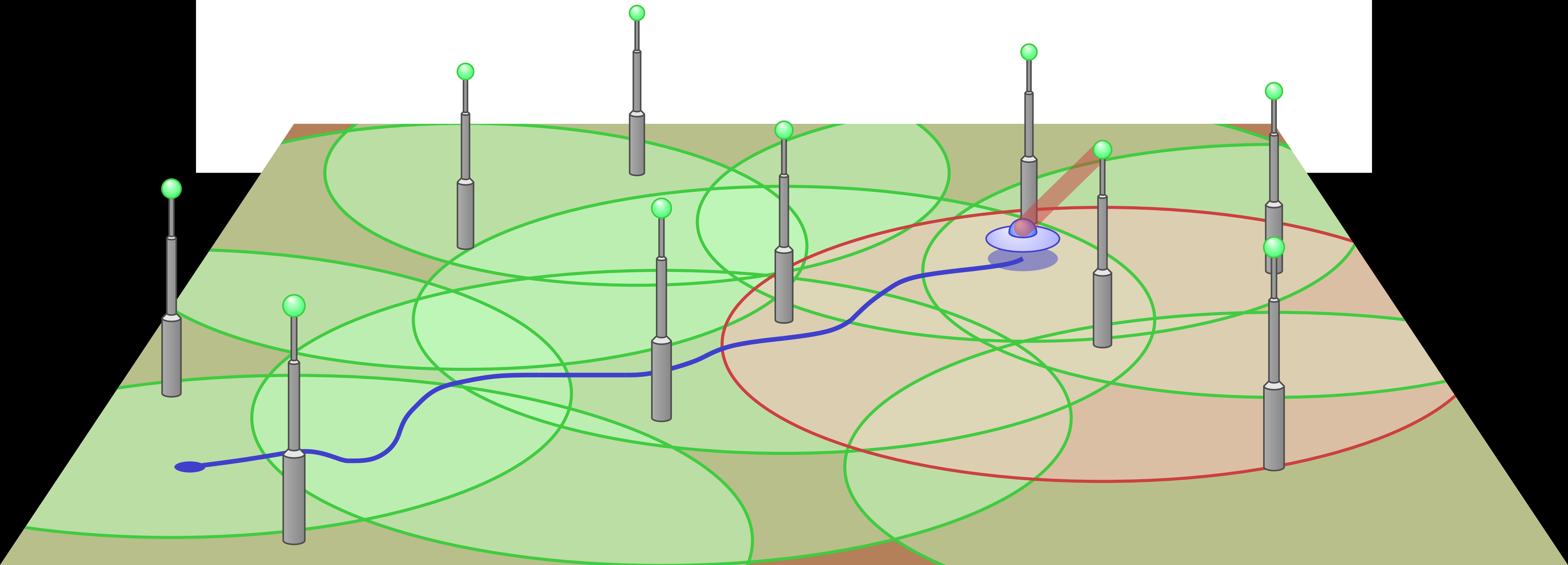
OBJECTIVES

- ~~Trilateration~~ c-lateration
  - UMO must be tracked by ~~3~~ c stations at every point in time
  - Let's focus on $c = 1$
- Handovers
  - When UMO leaves sensor region, we need to assign it to a different sensor



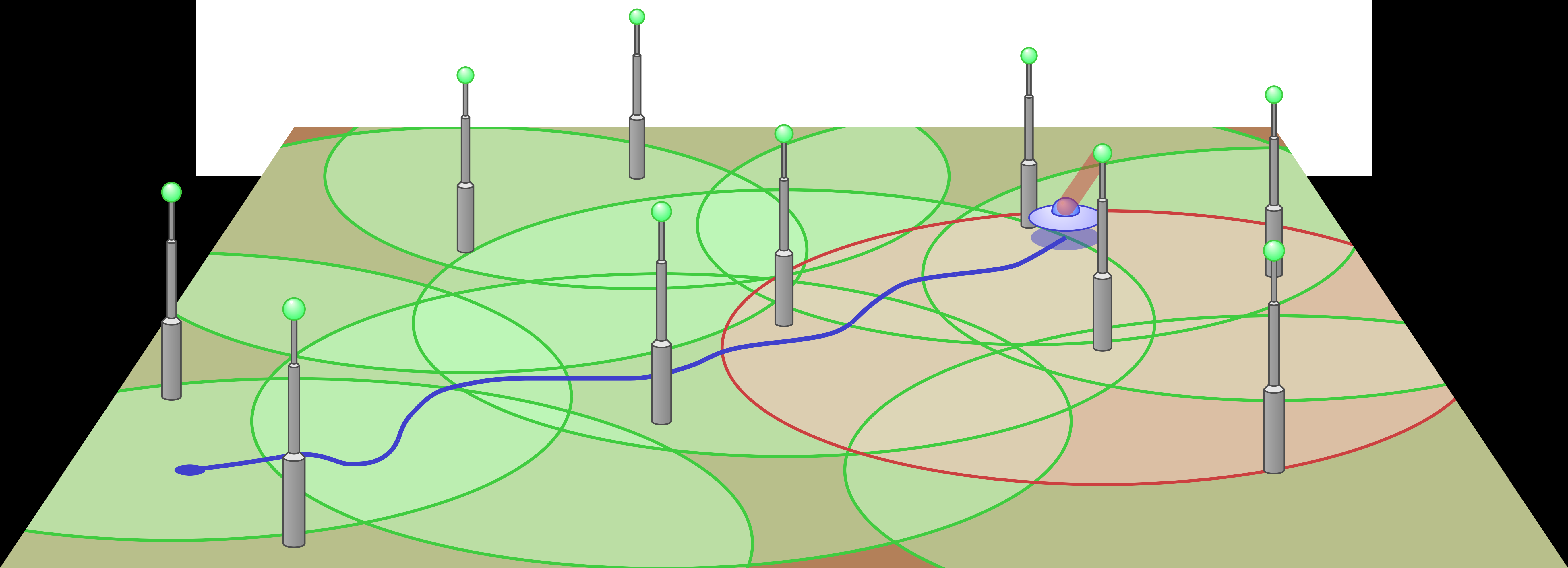
OBJECTIVES

- ~~Trilateration~~ c-lateration
  - UMO must be tracked by ~~3~~ c stations at every point in time
  - Let's focus on $c = 1$
- Handovers
  - When UMO leaves sensor region, we need to assign it to a different sensor



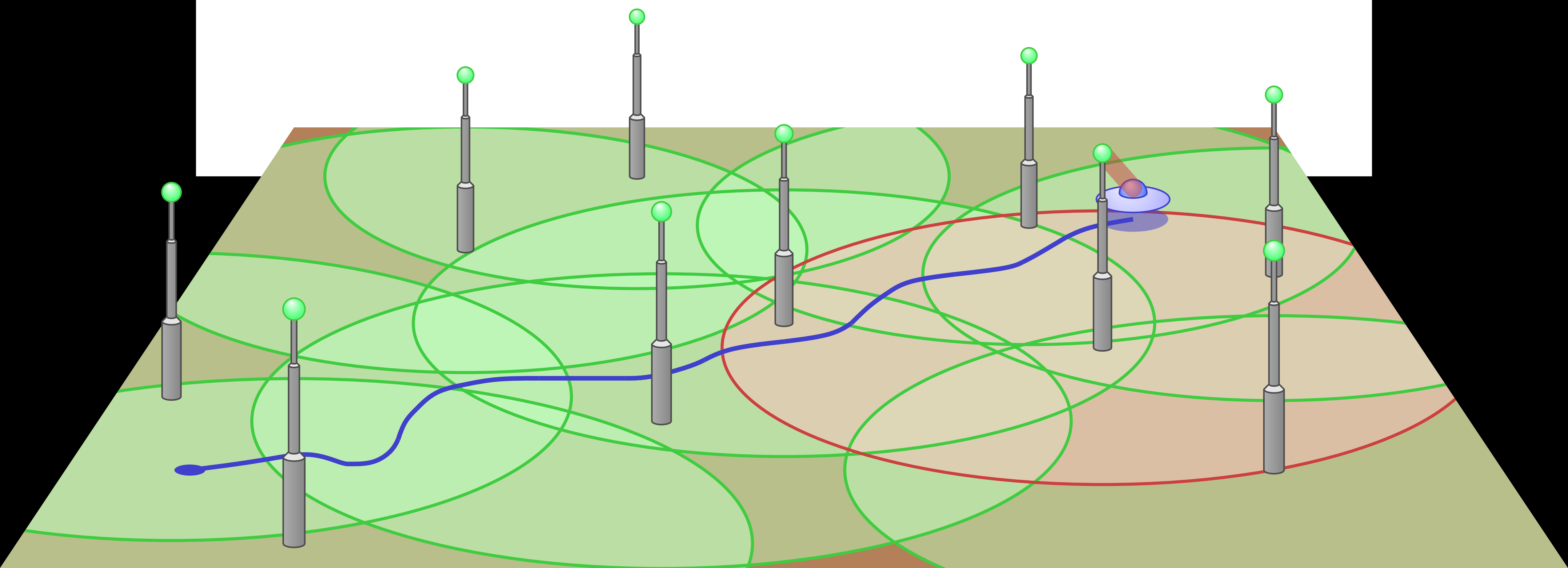
OBJECTIVES

- ~~Trilateration~~ c-lateration
  - UMO must be tracked by ~~3~~ c stations at every point in time
  - Let's focus on $c = 1$
- Handovers
  - When UMO leaves sensor region, we need to assign it to a different sensor



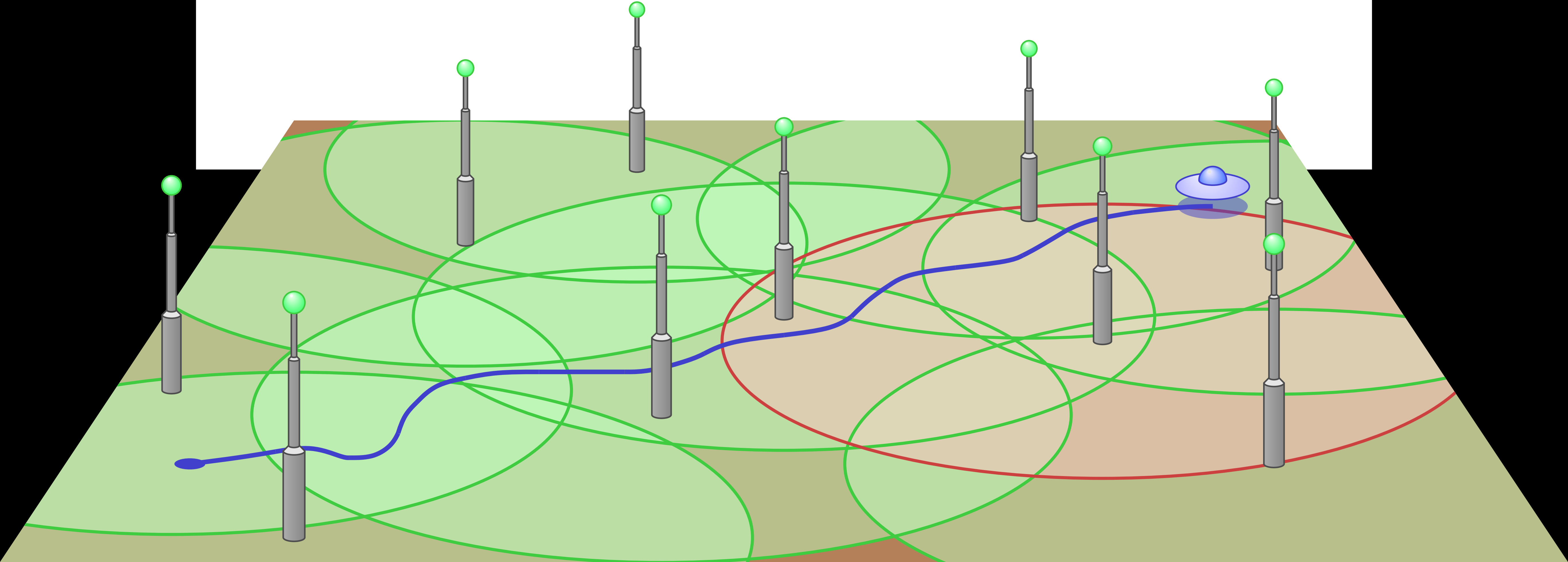
OBJECTIVES

- ~~Trilateration~~ c-lateration
  - UMO must be tracked by ~~3~~ c stations at every point in time
  - Let's focus on $c = 1$
- Handovers
  - When UMO leaves sensor region, we need to assign it to a different sensor



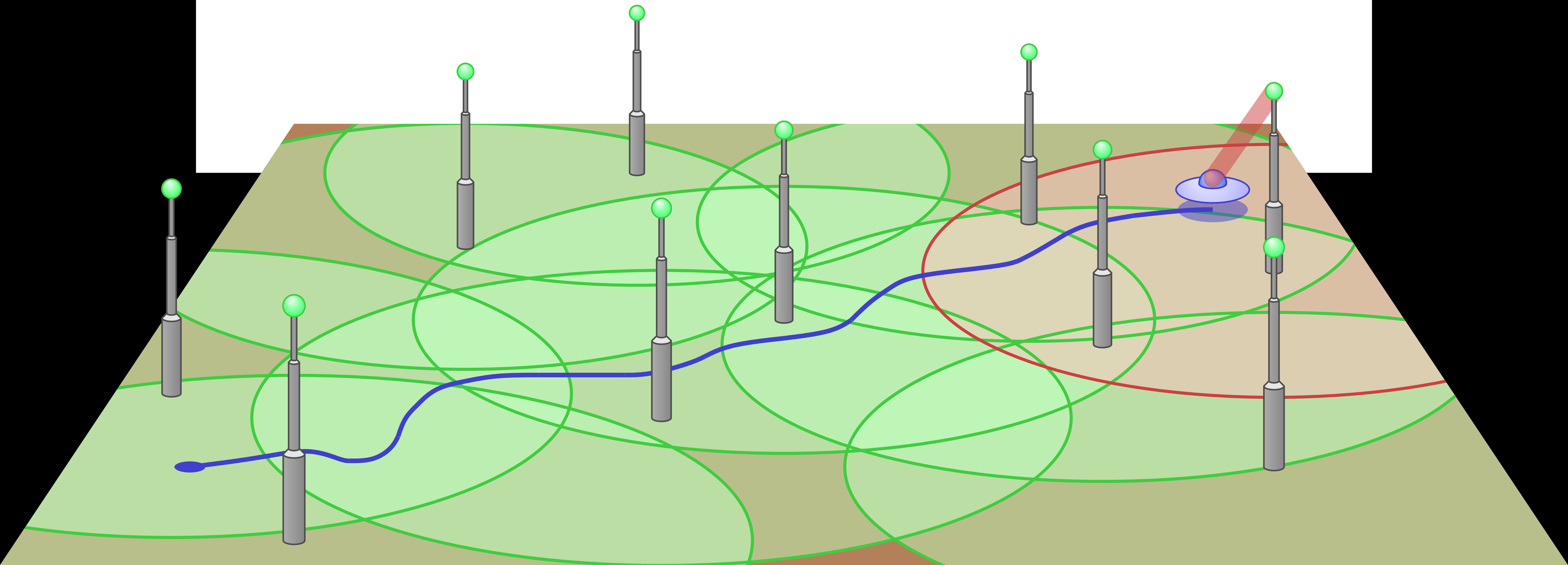
OBJECTIVES

- ~~Trilateration~~ c-lateration
  - UMO must be tracked by ~~3~~ c stations at every point in time
  - Let's focus on $c = 1$
- Handovers
  - When UMO leaves sensor region, we need to assign it to a different sensor



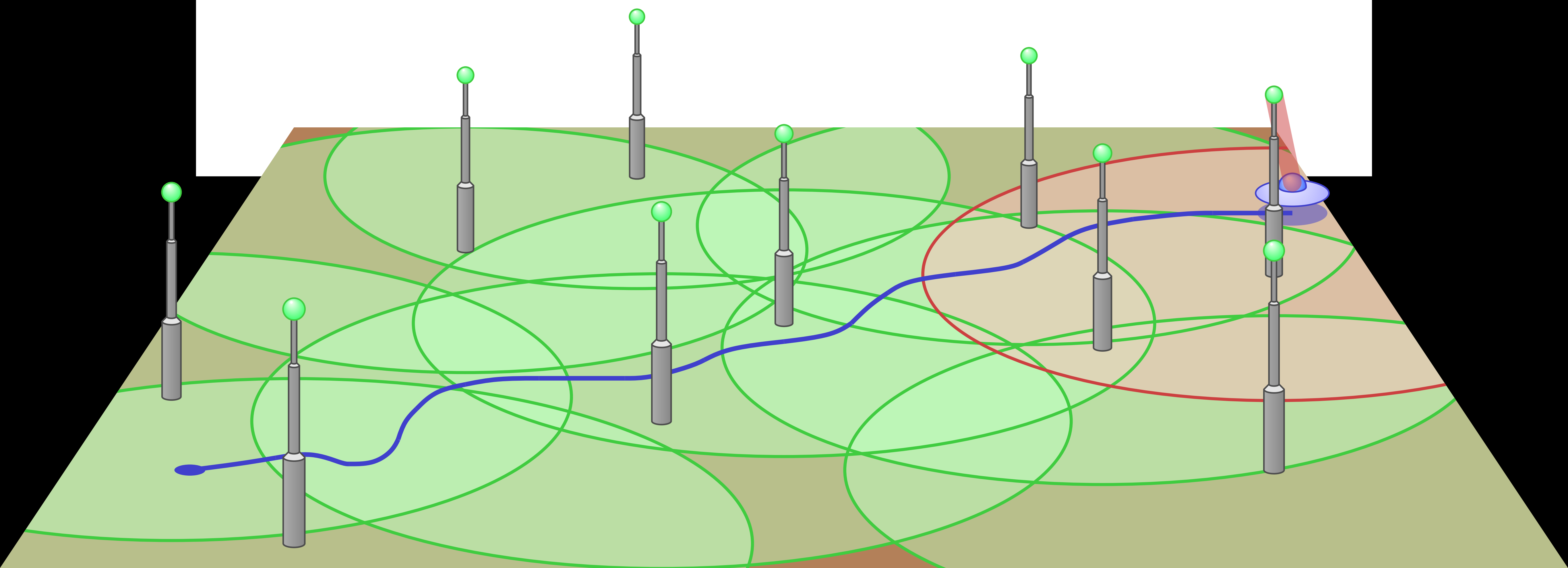
OBJECTIVES

- ~~Trilateration~~ c-lateration
  - UMO must be tracked by ~~3~~ c stations at every point in time
  - Let's focus on $c = 1$
- Handovers
  - When UMO leaves sensor region, we need to assign it to a different sensor



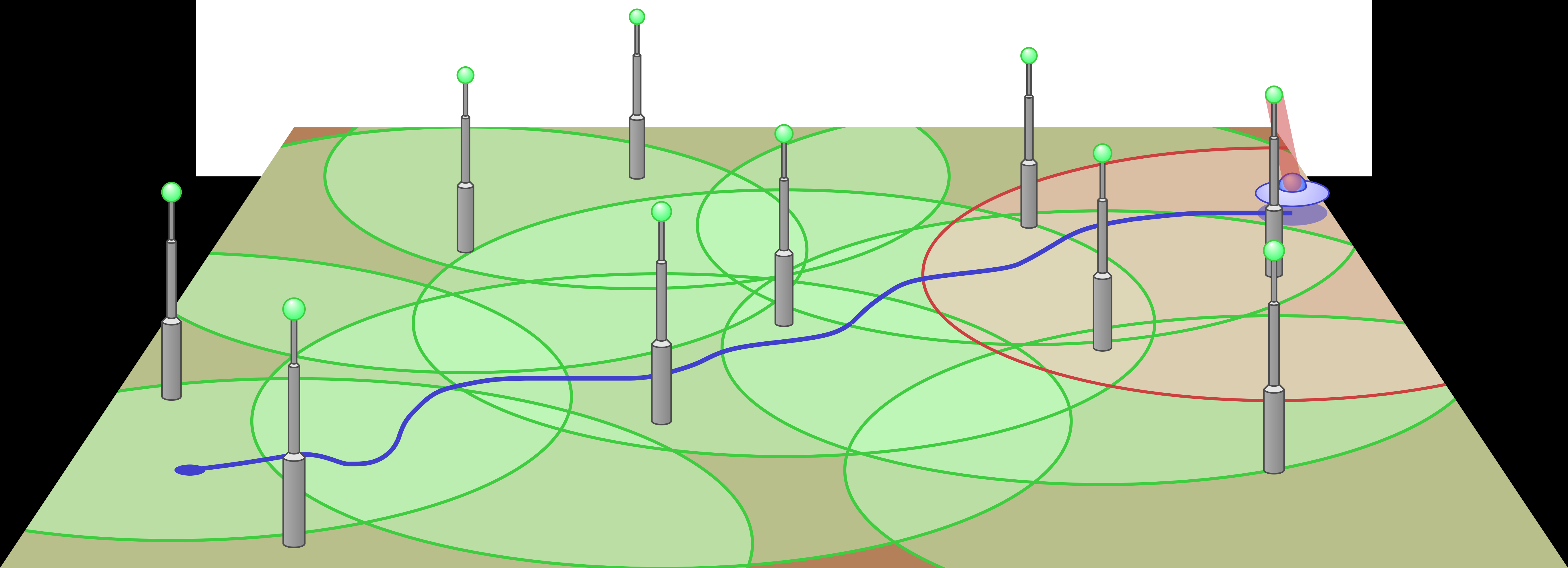
OBJECTIVES

- ~~Trilateration~~ c-lateration
  - UMO must be tracked by ~~3~~ c stations at every point in time
  - Let's focus on $c = 1$
- Handovers
  - When UMO leaves sensor region, we need to assign it to a different sensor



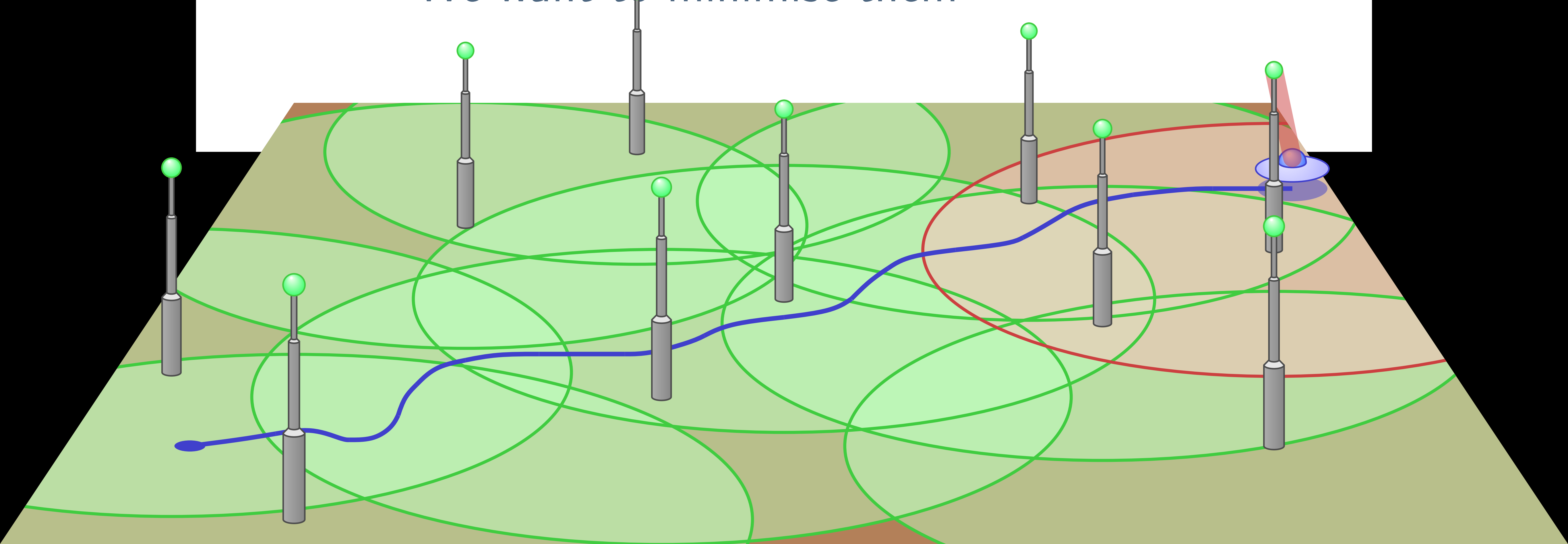
OBJECTIVES

- ~~Trilateration~~ c-lateration
  - UMO must be tracked by ~~3~~ c stations at every point in time
  - Let's focus on $c = 1$
- Handovers
  - When UMO leaves sensor region, we need to assign it to a different sensor
  - Handovers are costly



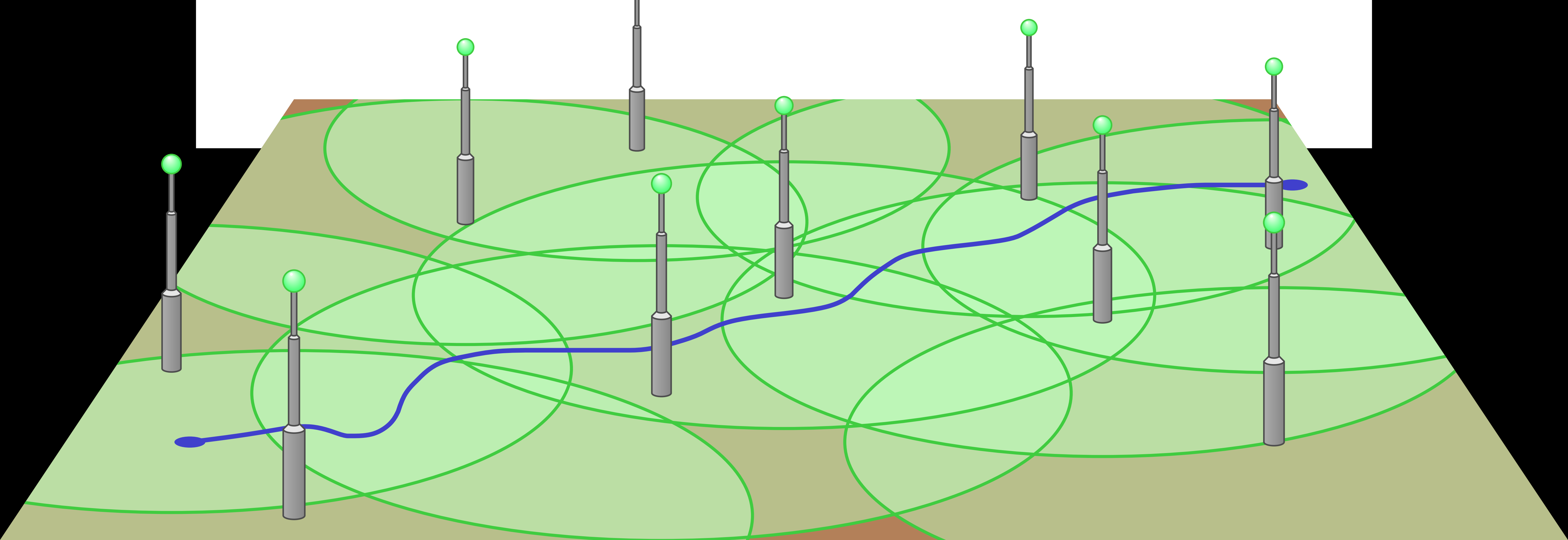
OBJECTIVES

- ~~Trilateration~~ c-lateration
  - UMO must be tracked by ~~3~~ c stations at every point in time
  - Let's focus on $c = 1$
- Handovers
  - When UMO leaves sensor region, we need to assign it to a different sensor
  - Handovers are costly
  - We want to minimise them



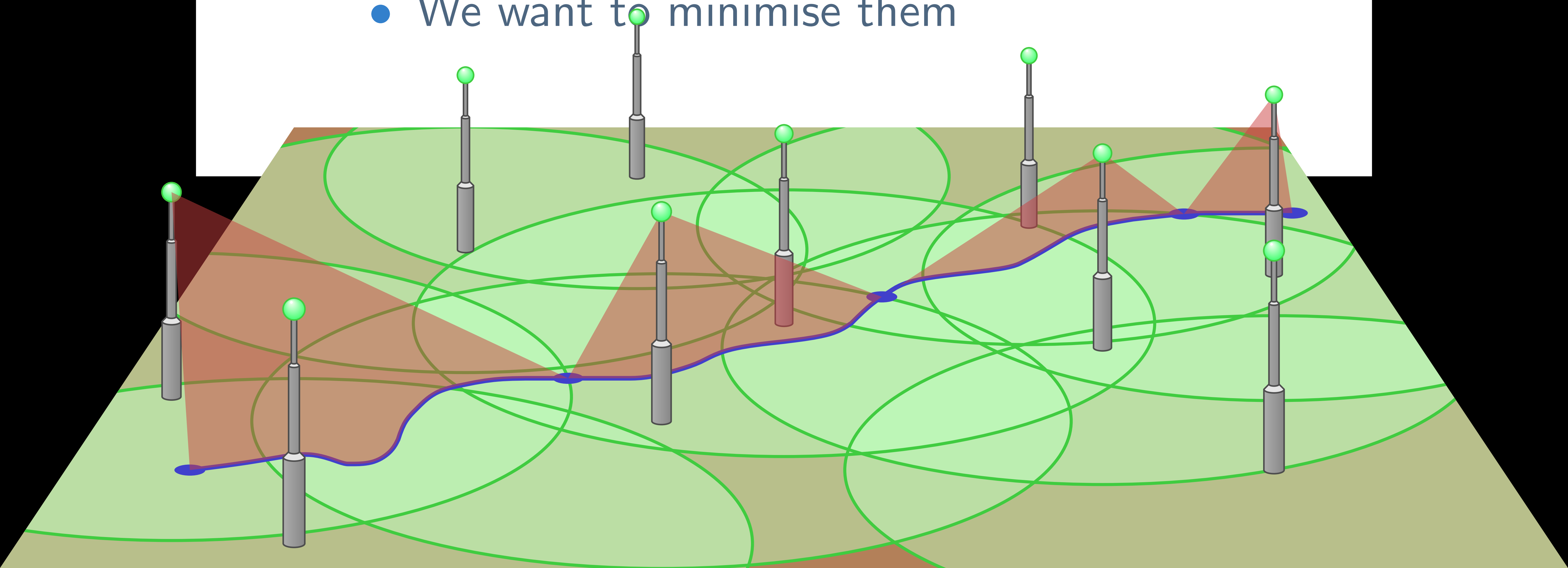
OBJECTIVES

- ~~Trilateration~~ c-lateration
  - UMO must be tracked by ~~3~~ c stations at every point in time
  - Let's focus on $c = 1$
- Handovers
  - When UMO leaves sensor region, we need to assign it to a different sensor
  - Handovers are costly
  - We want to minimise them



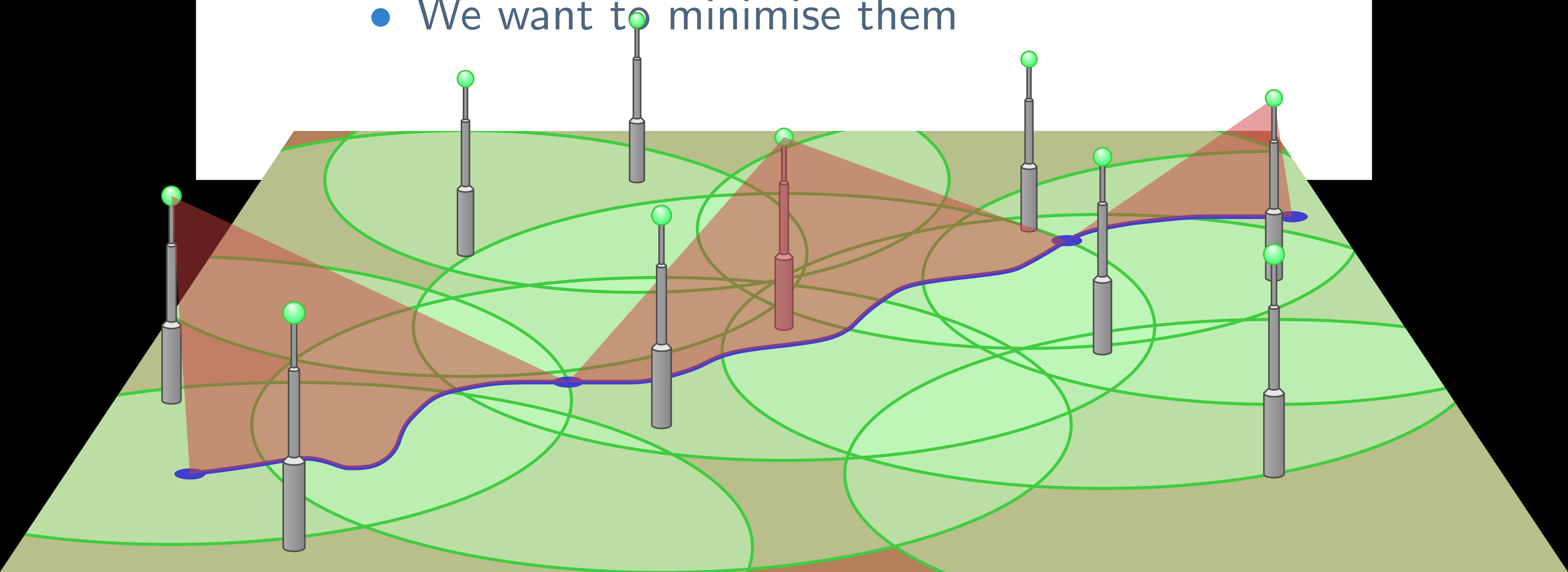
OBJECTIVES

- ~~Trilateration~~ c-lateration
  - UMO must be tracked by ~~3~~ c stations at every point in time
  - Let's focus on $c = 1$
- Handovers
  - When UMO leaves sensor region, we need to assign it to a different sensor
  - Handovers are costly
  - We want to minimise them



OBJECTIVES

- ~~Trilateration~~ c-lateration
  - UMO must be tracked by ~~3~~ c stations at every point in time
  - Let's focus on $c = 1$
- Handovers
  - When UMO leaves sensor region, we need to assign it to a different sensor
  - Handovers are costly
  - We want to minimise them



ONLINE ALGORITHM

ONLINE ALGORITHM

- Input

ONLINE ALGORITHM

- Input
  - Fixed set of regions in the plane

ONLINE ALGORITHM

- Input
  - Fixed set of regions in the plane
  - Sequence of locations for the UMO

ONLINE ALGORITHM

- Input
  - Fixed set of regions in the plane
  - Sequence of locations for the UMO
- Output

ONLINE ALGORITHM

- Input
  - Fixed set of regions in the plane
  - Sequence of locations for the UMO
- Output
  - Sequence of region assignments

ONLINE ALGORITHM

- Input
  - Fixed set of regions in the plane
  - Sequence of locations for the UMO
- Output
  - Sequence of region assignments
  - Algorithm needs to react to each event before knowing what the next event will be

ONLINE ALGORITHM

- Input
  - Fixed set of regions in the plane
  - Sequence of locations for the UMO
- Output
  - Sequence of region assignments
  - Algorithm needs to react to each event before knowing what the next event will be
- Performance measure

ONLINE ALGORITHM

- Input
  - Fixed set of regions in the plane
  - Sequence of locations for the UMO
- Output
  - Sequence of region assignments
  - Algorithm needs to react to each event before knowing what the next event will be
- Performance measure
  - *Competitive ratio*: cost of online algorithm divided by cost of optimal offline algorithm

ONLINE ALGORITHM TYPES

ONLINE ALGORITHM TYPES

- Stateless algorithms

ONLINE ALGORITHM TYPES

- Stateless algorithms
  - Algorithm knows only about current event

ONLINE ALGORITHM TYPES

- Stateless algorithms
  - Algorithm knows only about current event
  - Adversary knows everything

ONLINE ALGORITHM TYPES

- Stateless algorithms
  - Algorithm knows only about current event
  - Adversary knows everything
- Deterministic algorithms

ONLINE ALGORITHM TYPES

- Stateless algorithms
  - Algorithm knows only about current event
  - Adversary knows everything
- Deterministic algorithms
  - Algorithm knows all past events

ONLINE ALGORITHM TYPES

- Stateless algorithms
  - Algorithm knows only about current event
  - Adversary knows everything
- Deterministic algorithms
  - Algorithm knows all past events
  - Adversary knows everything

ONLINE ALGORITHM TYPES

- Stateless algorithms
  - Algorithm knows only about current event
  - Adversary knows everything
- Deterministic algorithms
  - Algorithm knows all past events
  - Adversary knows everything
- Randomised algorithms

ONLINE ALGORITHM TYPES

- Stateless algorithms
  - Algorithm knows only about current event
  - Adversary knows everything
- Deterministic algorithms
  - Algorithm knows all past events
  - Adversary knows everything
- Randomised algorithms
  - Algorithm knows all past events

ONLINE ALGORITHM TYPES

- Stateless algorithms
  - Algorithm knows only about current event
  - Adversary knows everything
- Deterministic algorithms
  - Algorithm knows all past events
  - Adversary knows everything
- Randomised algorithms
  - Algorithm knows all past events
  - Adversary knows everything except the random choices made by the algorithm

FIRST OBSERVATION. . .

FIRST OBSERVATION...

- Competitive ratio can be pretty bad

FIRST OBSERVATION...

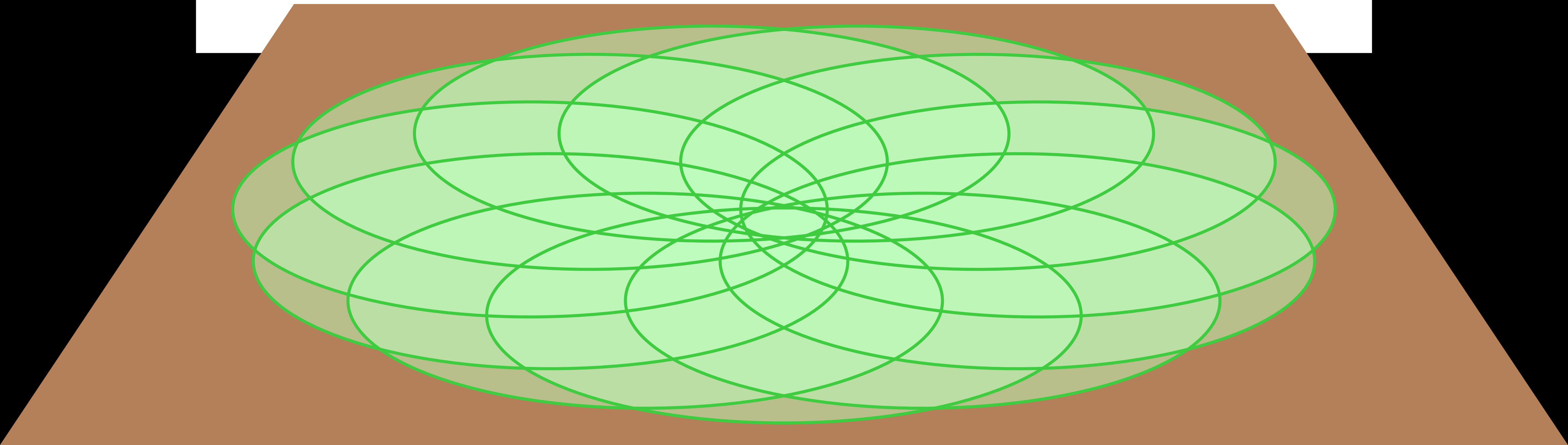
- Competitive ratio can be pretty bad
  - Let all disks have a common interior

FIRST OBSERVATION...

- Competitive ratio can be pretty bad
  - Let all disks have a common interior

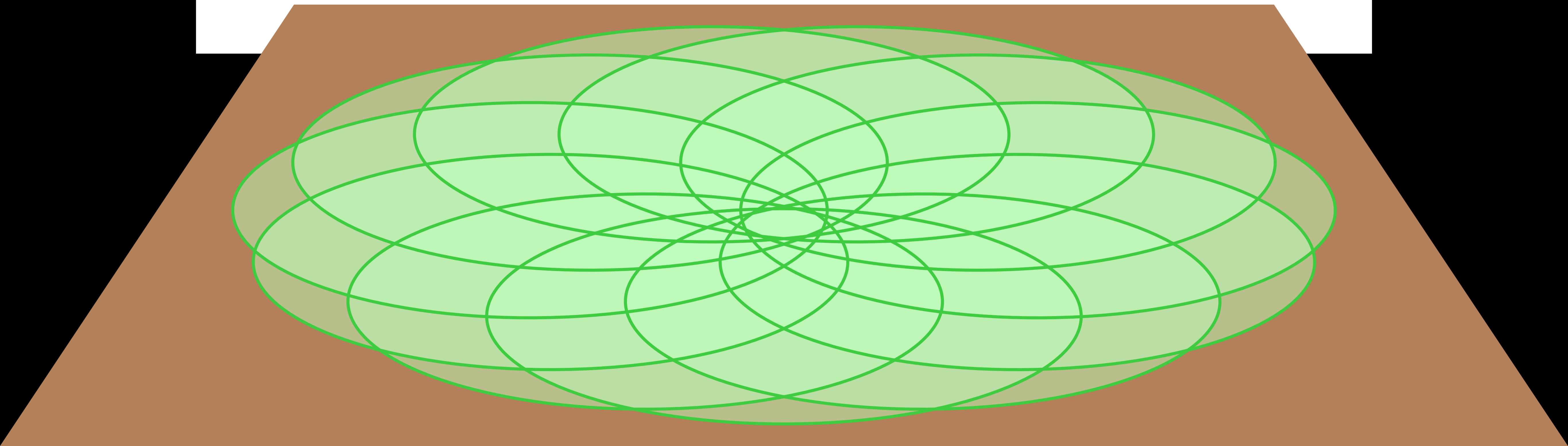
FIRST OBSERVATION...

- Competitive ratio can be pretty bad
  - Let all disks have a common interior



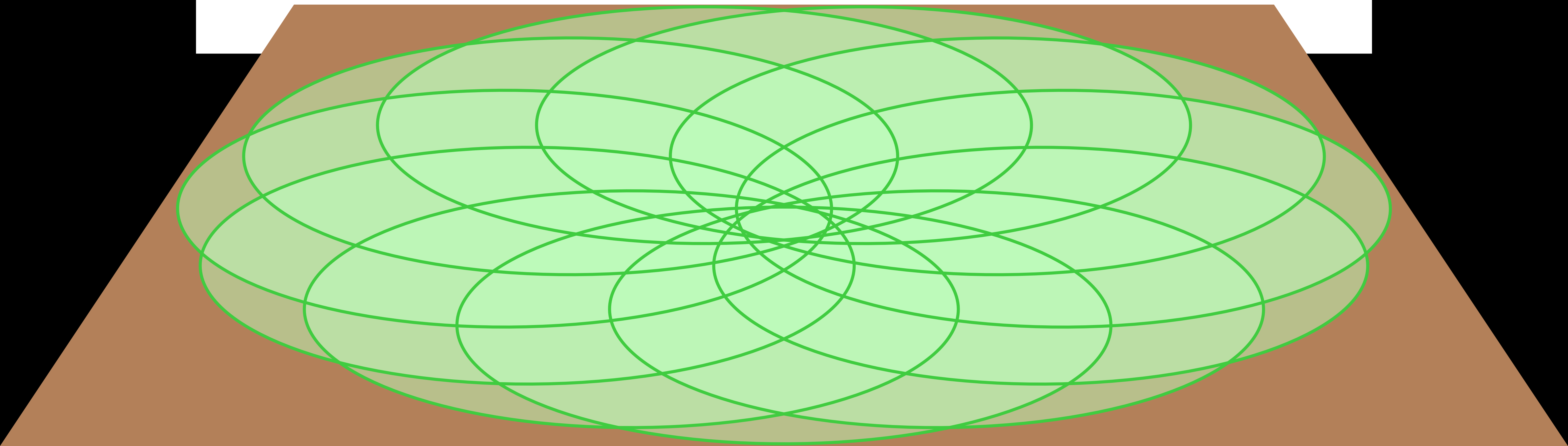
FIRST OBSERVATION...

- Competitive ratio can be pretty bad
  - Let all disks have a common interior
  - Zoom in on the center



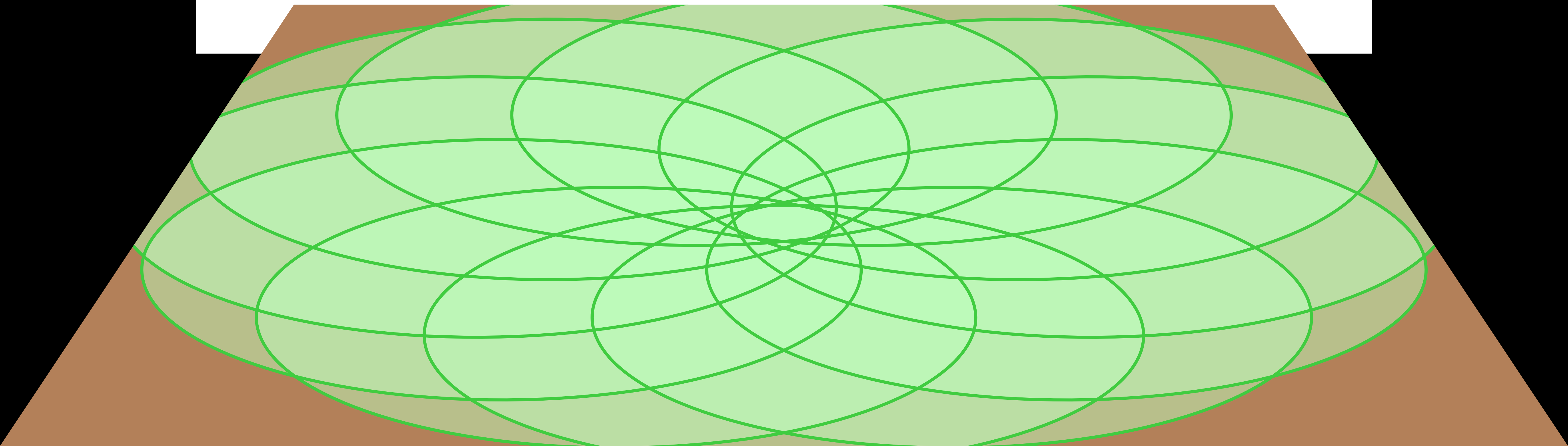
FIRST OBSERVATION...

- Competitive ratio can be pretty bad
  - Let all disks have a common interior
  - Zoom in on the center



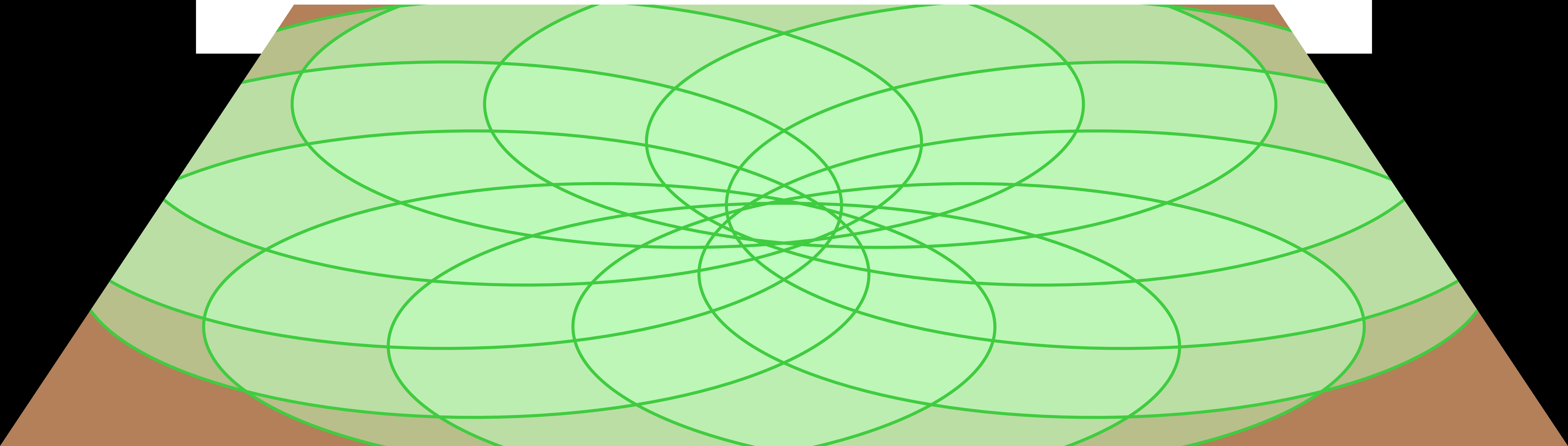
FIRST OBSERVATION...

- Competitive ratio can be pretty bad
  - Let all disks have a common interior
  - Zoom in on the center



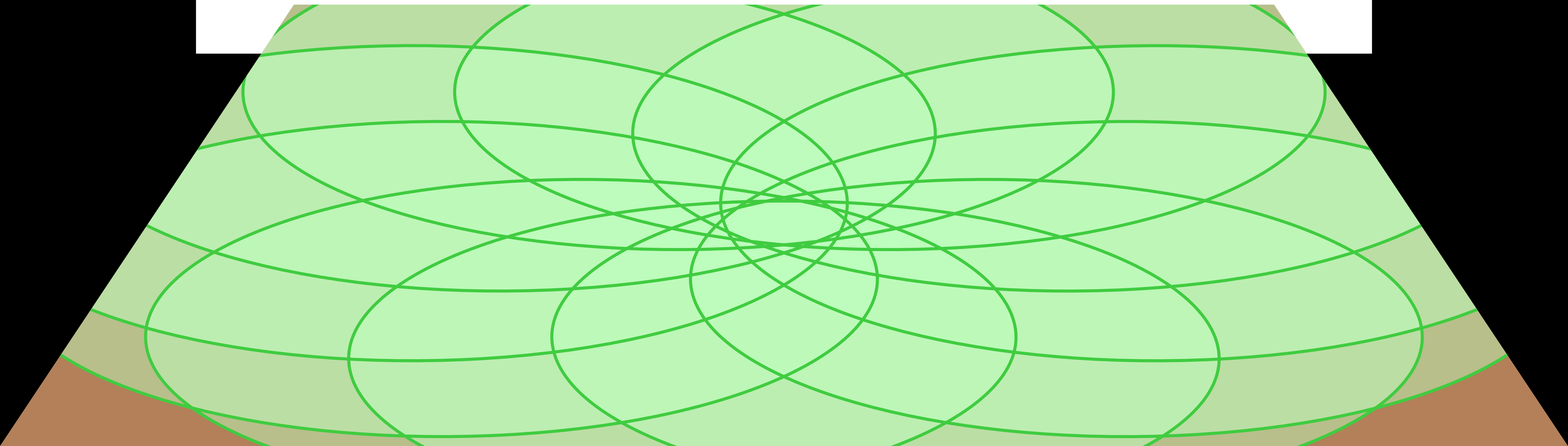
FIRST OBSERVATION...

- Competitive ratio can be pretty bad
  - Let all disks have a common interior
  - Zoom in on the center



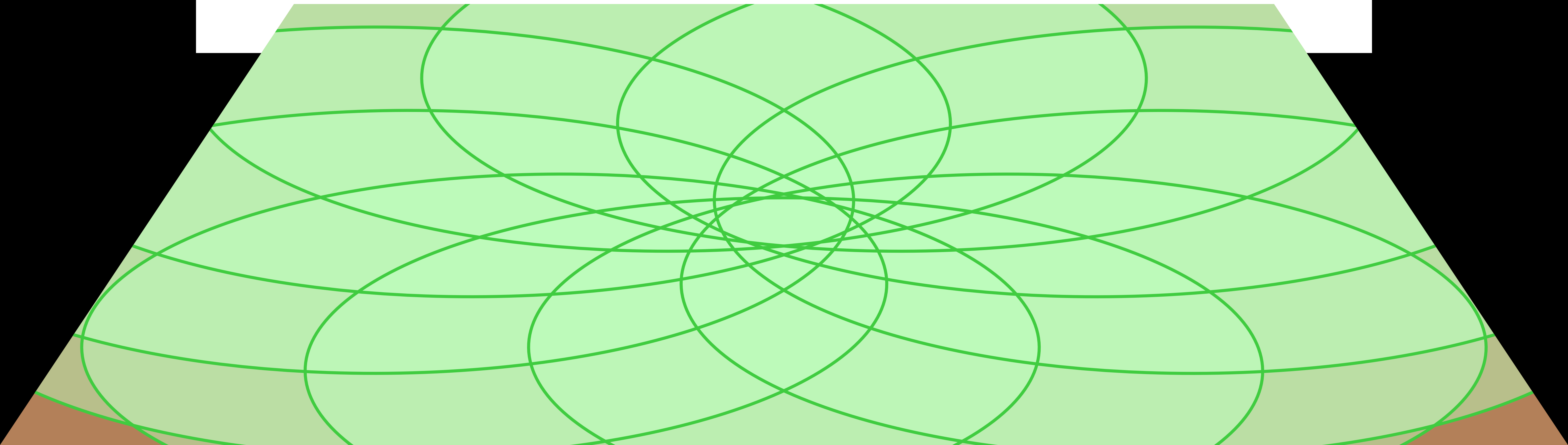
FIRST OBSERVATION...

- Competitive ratio can be pretty bad
  - Let all disks have a common interior
  - Zoom in on the center



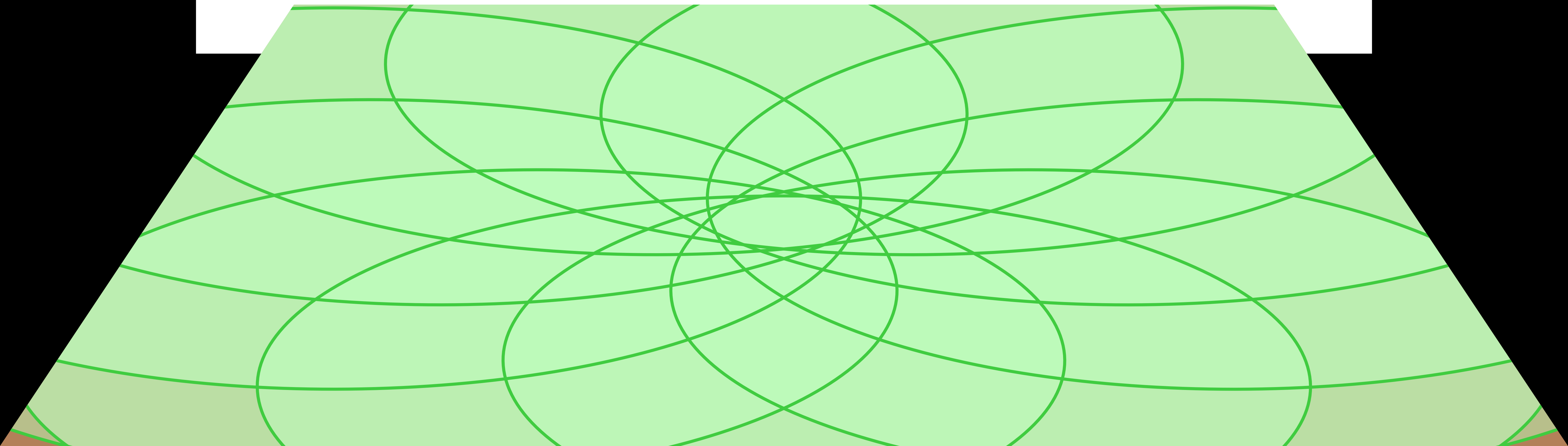
FIRST OBSERVATION...

- Competitive ratio can be pretty bad
  - Let all disks have a common interior
  - Zoom in on the center



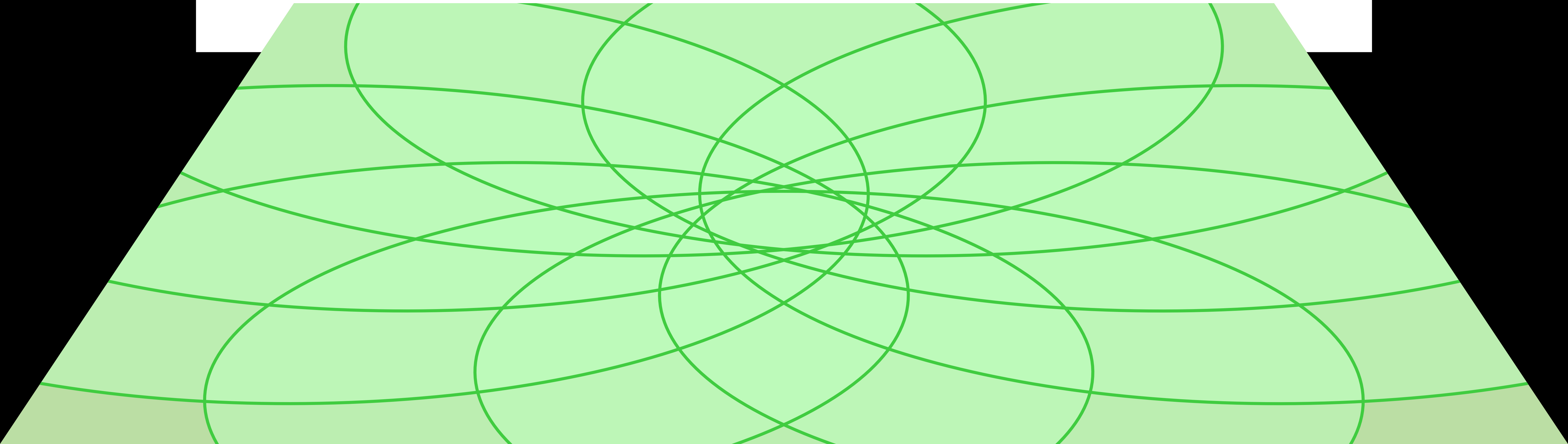
FIRST OBSERVATION...

- Competitive ratio can be pretty bad
  - Let all disks have a common interior
  - Zoom in on the center



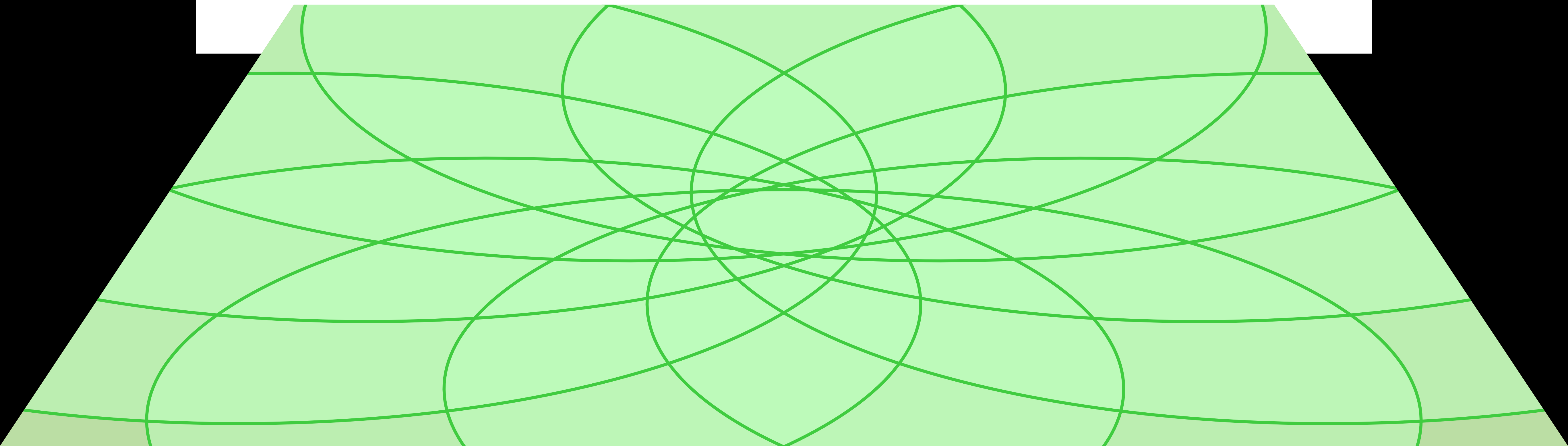
FIRST OBSERVATION…

- Competitive ratio can be pretty bad
  - Let all disks have a common interior
  - Zoom in on the center



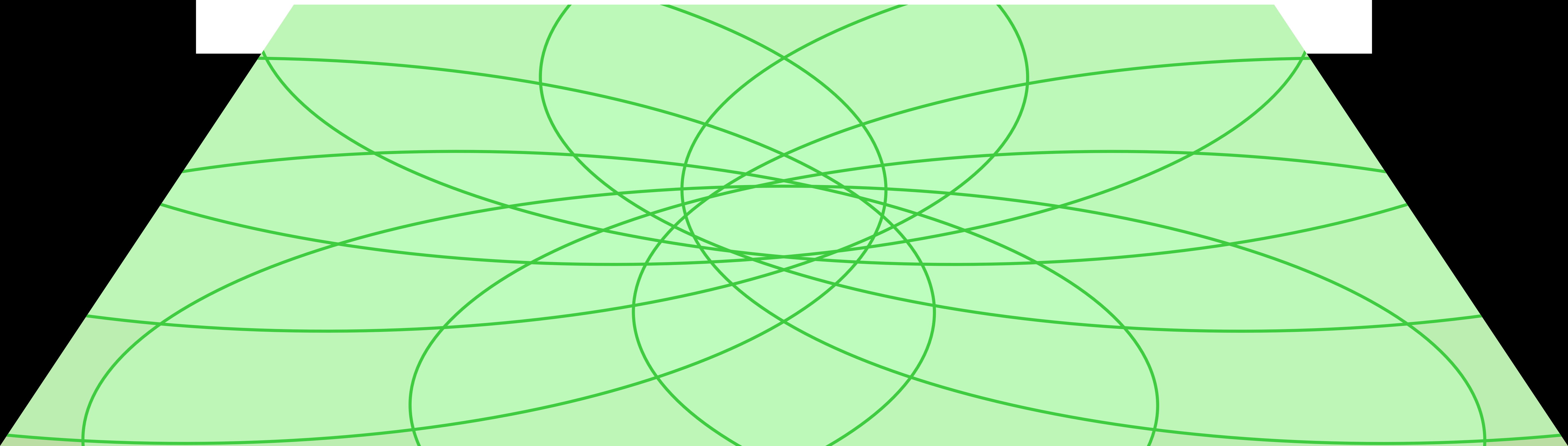
FIRST OBSERVATION...

- Competitive ratio can be pretty bad
  - Let all disks have a common interior
  - Zoom in on the center



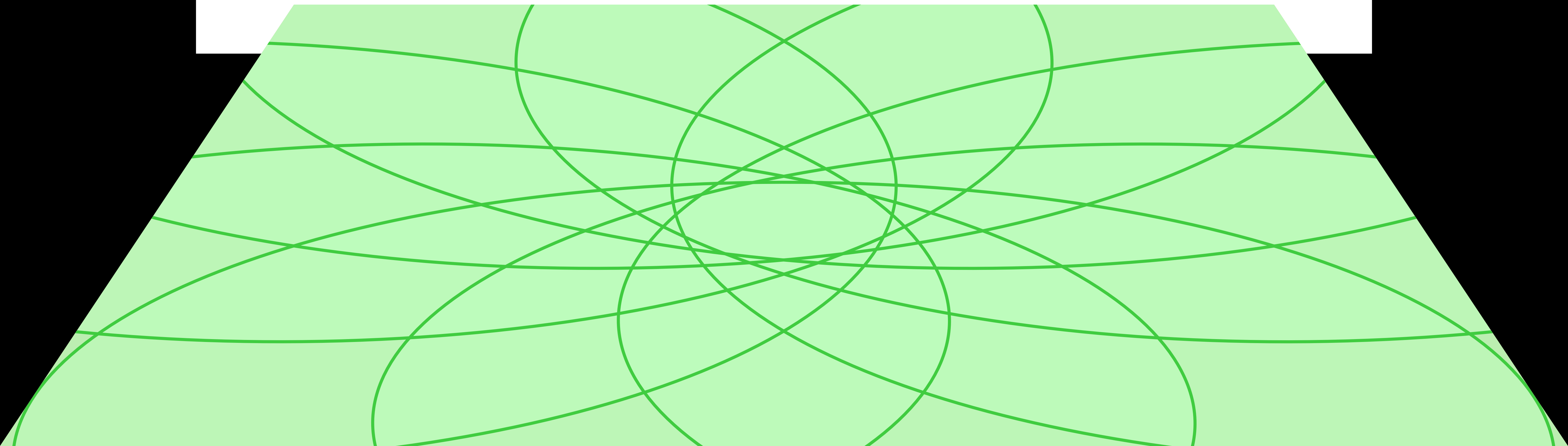
FIRST OBSERVATION...

- Competitive ratio can be pretty bad
  - Let all disks have a common interior
  - Zoom in on the center



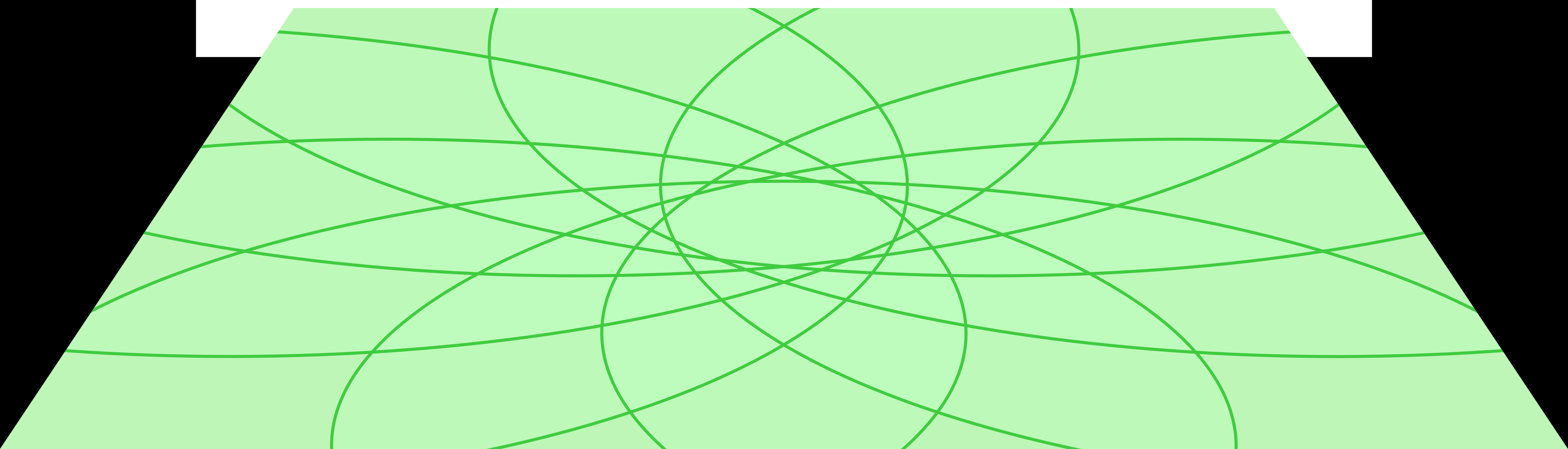
FIRST OBSERVATION...

- Competitive ratio can be pretty bad
  - Let all disks have a common interior
  - Zoom in on the center



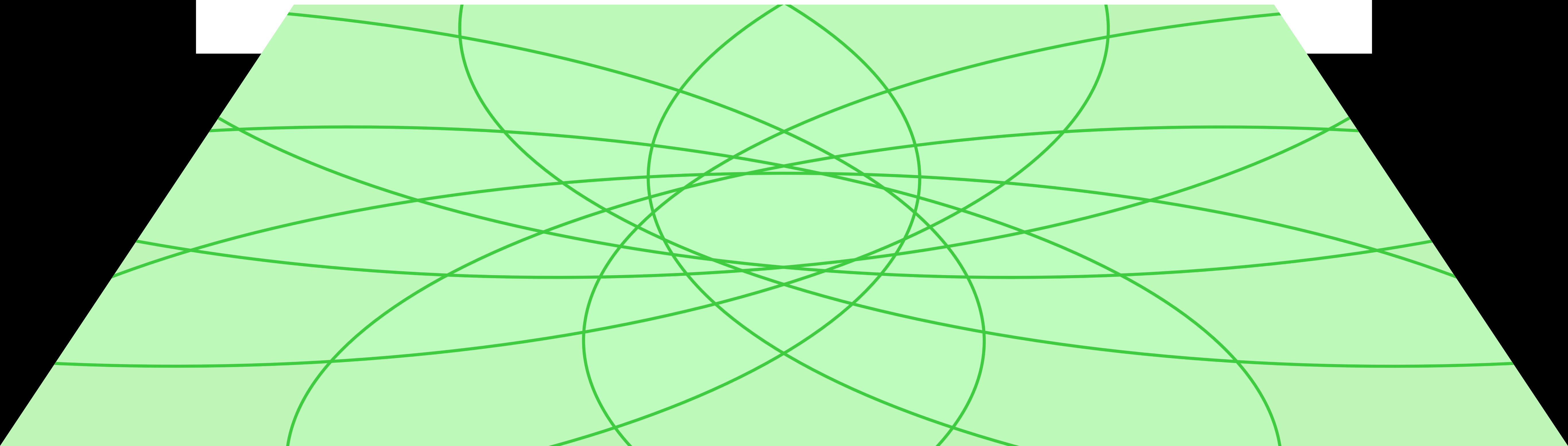
FIRST OBSERVATION...

- Competitive ratio can be pretty bad
  - Let all disks have a common interior
  - Zoom in on the center



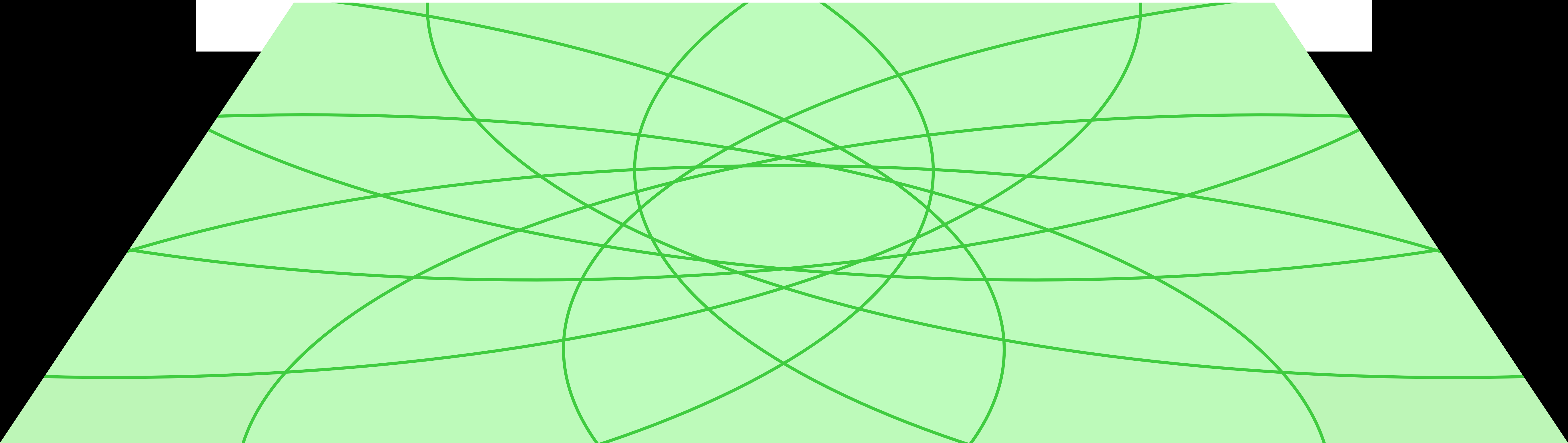
FIRST OBSERVATION...

- Competitive ratio can be pretty bad
  - Let all disks have a common interior
  - Zoom in on the center



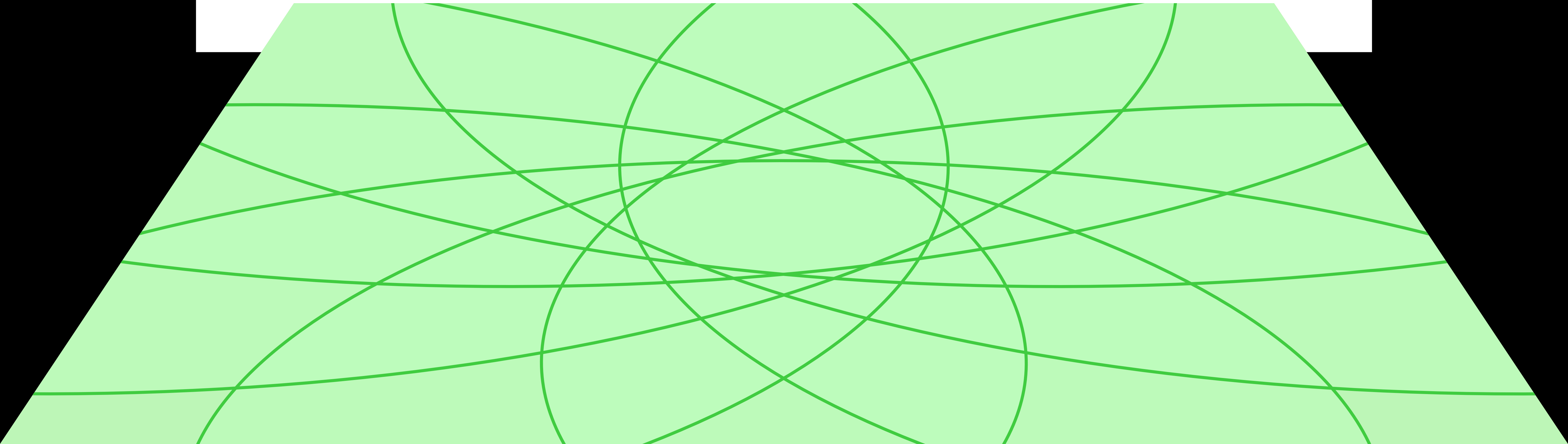
FIRST OBSERVATION...

- Competitive ratio can be pretty bad
  - Let all disks have a common interior
  - Zoom in on the center



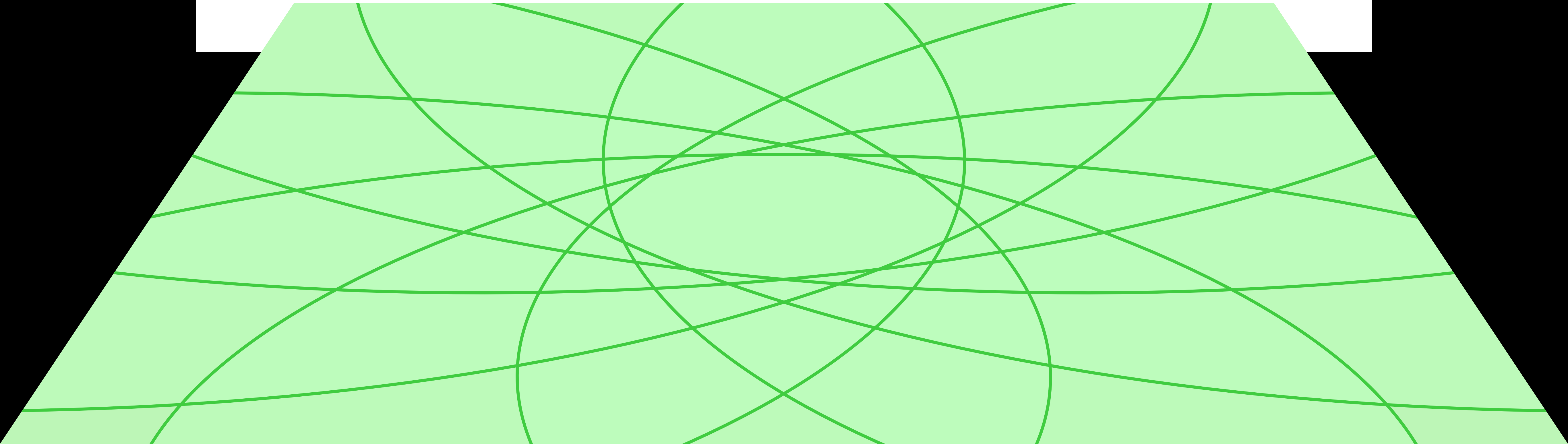
FIRST OBSERVATION...

- Competitive ratio can be pretty bad
  - Let all disks have a common interior
  - Zoom in on the center



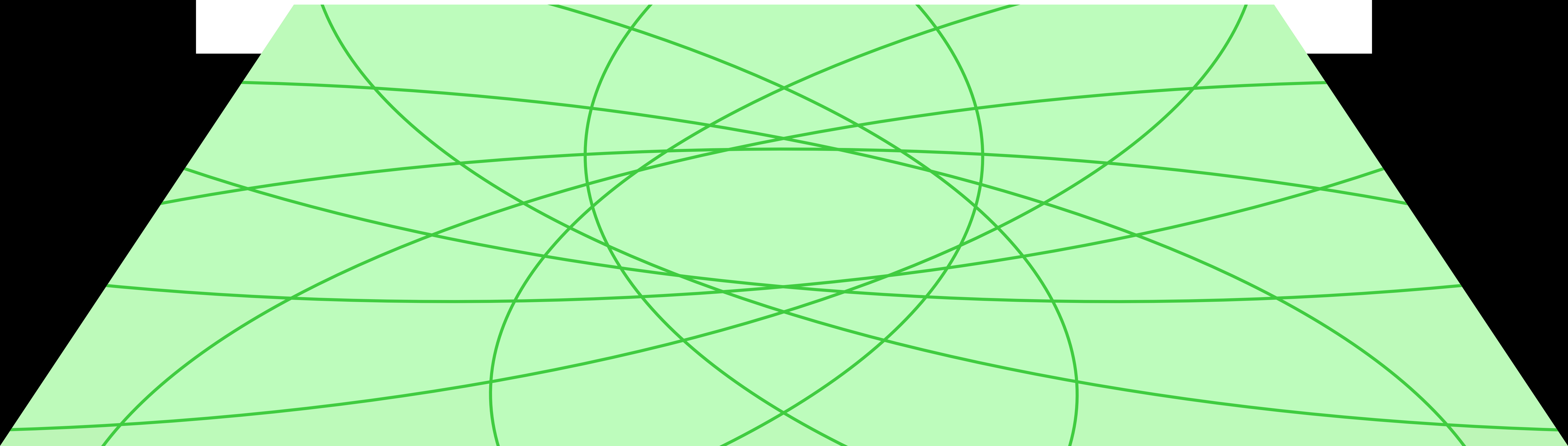
FIRST OBSERVATION...

- Competitive ratio can be pretty bad
  - Let all disks have a common interior
  - Zoom in on the center



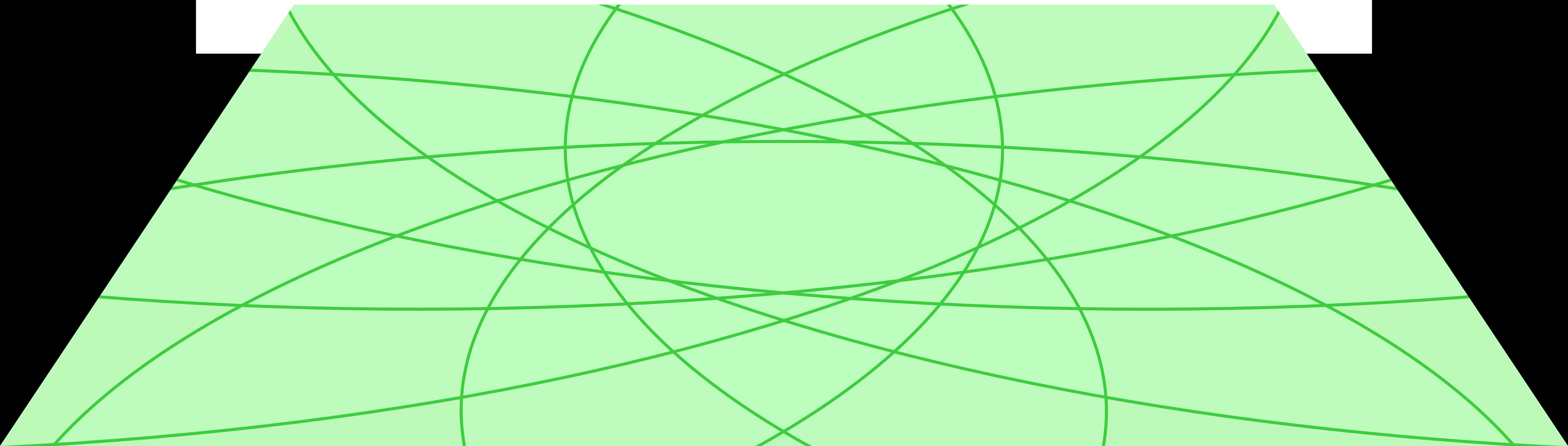
FIRST OBSERVATION...

- Competitive ratio can be pretty bad
  - Let all disks have a common interior
  - Zoom in on the center



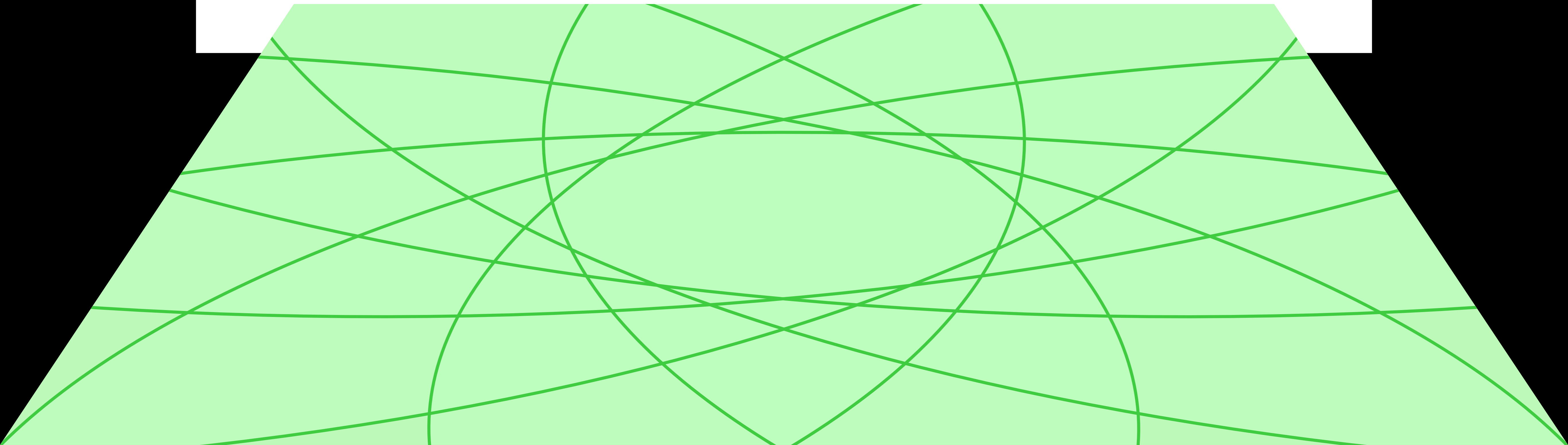
FIRST OBSERVATION…

- Competitive ratio can be pretty bad
  - Let all disks have a common interior
  - Zoom in on the center



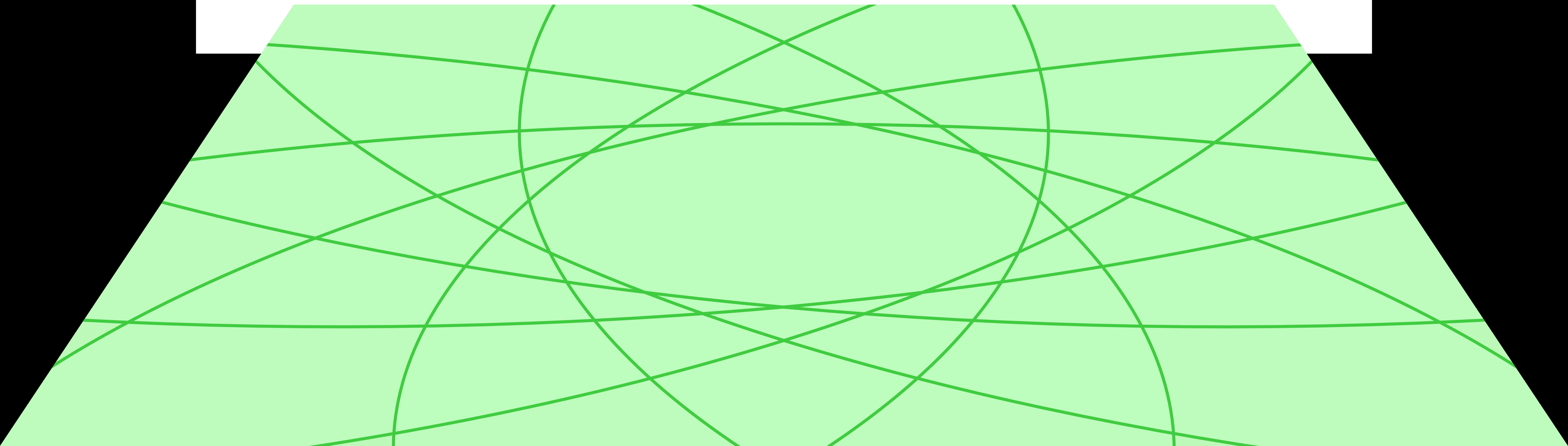
FIRST OBSERVATION...

- Competitive ratio can be pretty bad
  - Let all disks have a common interior
  - Zoom in on the center



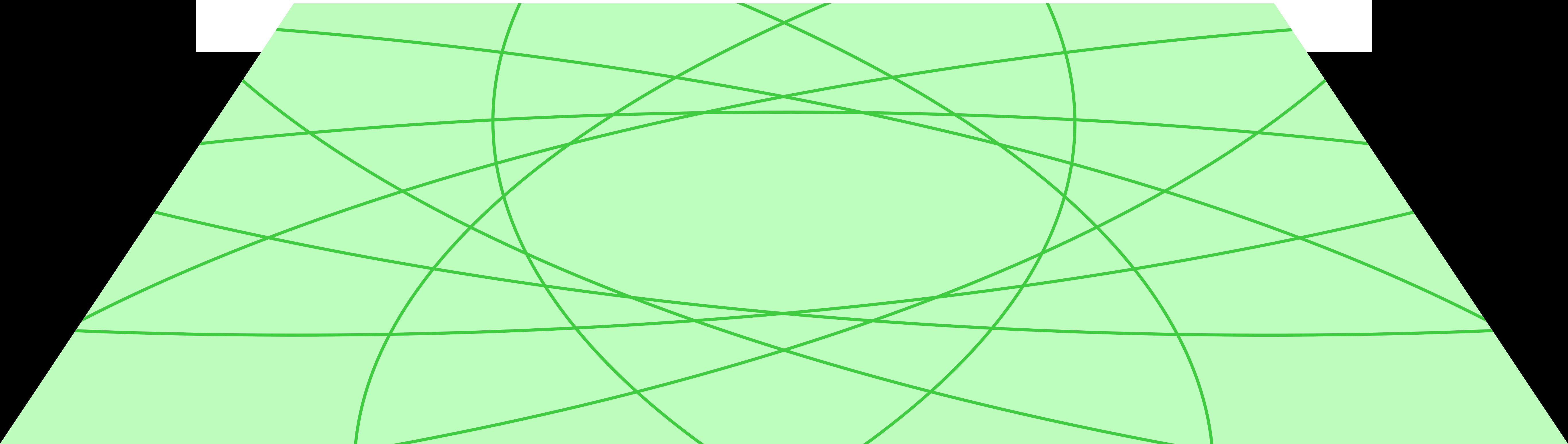
FIRST OBSERVATION...

- Competitive ratio can be pretty bad
  - Let all disks have a common interior
  - Zoom in on the center



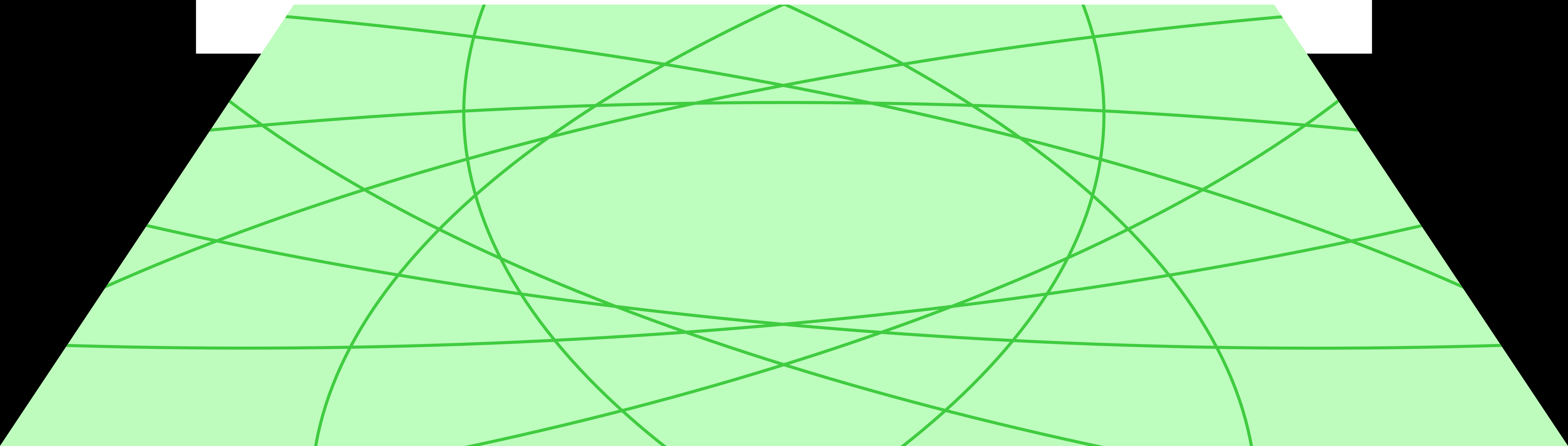
FIRST OBSERVATION…

- Competitive ratio can be pretty bad
  - Let all disks have a common interior
  - Zoom in on the center



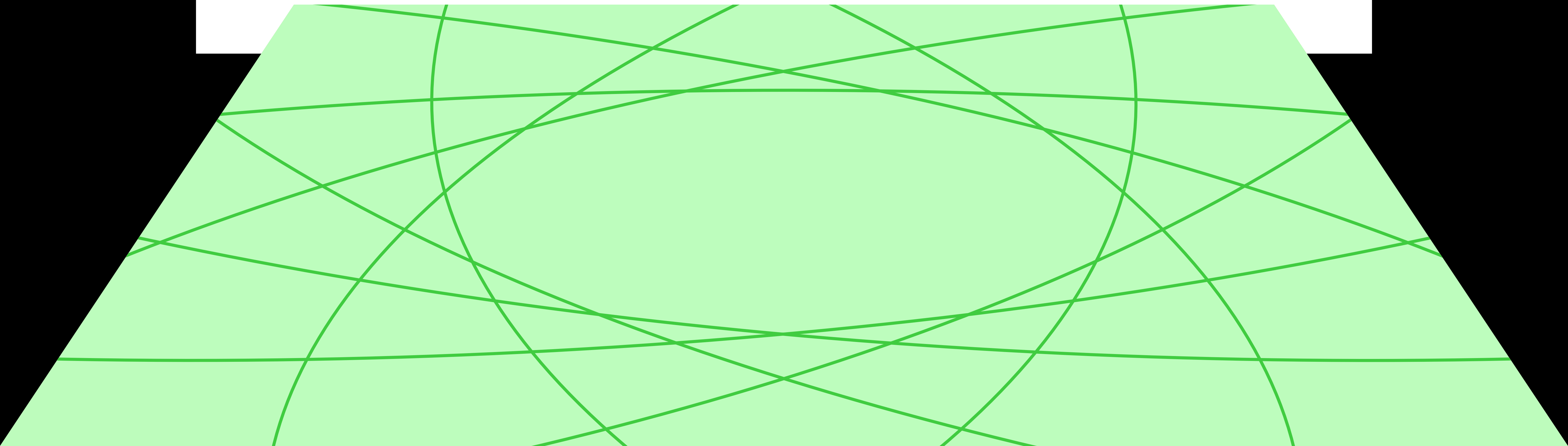
FIRST OBSERVATION...

- Competitive ratio can be pretty bad
  - Let all disks have a common interior
  - Zoom in on the center



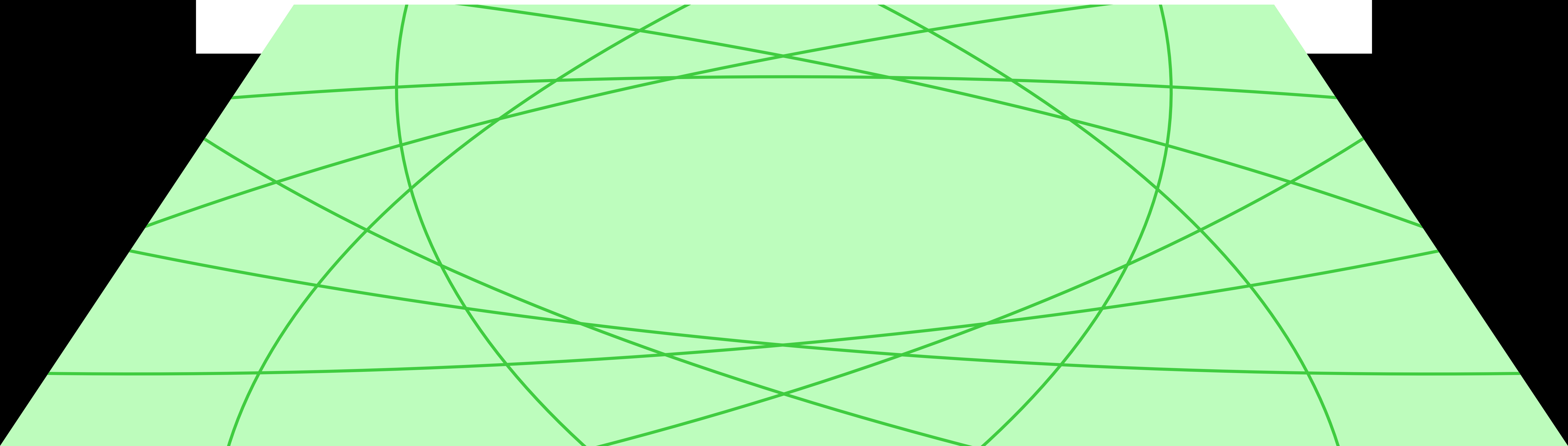
FIRST OBSERVATION...

- Competitive ratio can be pretty bad
  - Let all disks have a common interior
  - Zoom in on the center



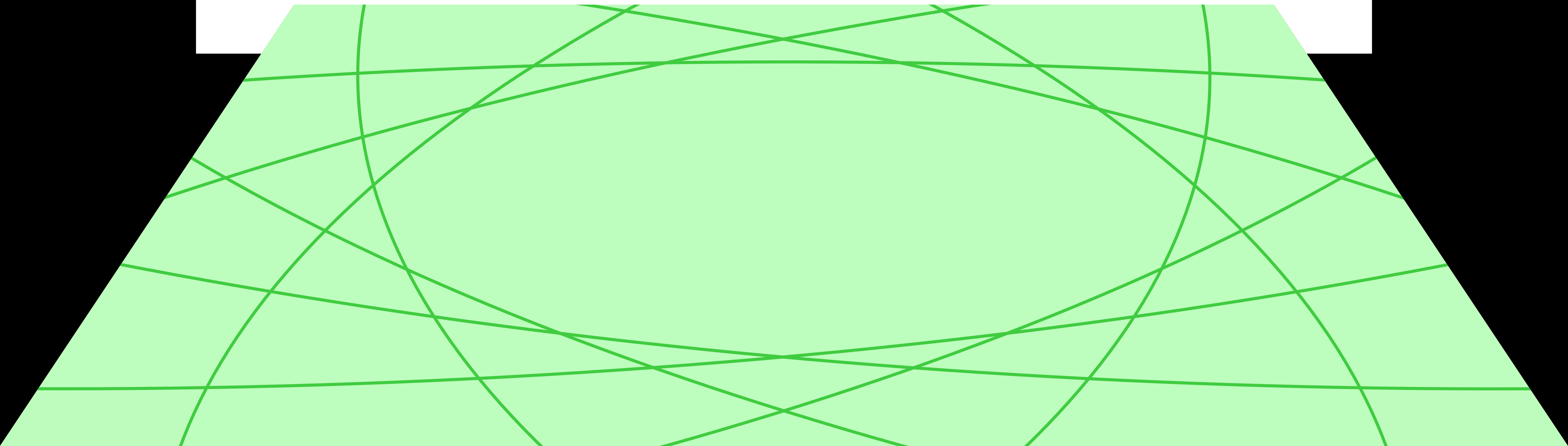
FIRST OBSERVATION. . .

- Competitive ratio can be pretty bad
  - Let all disks have a common interior
  - Zoom in on the center



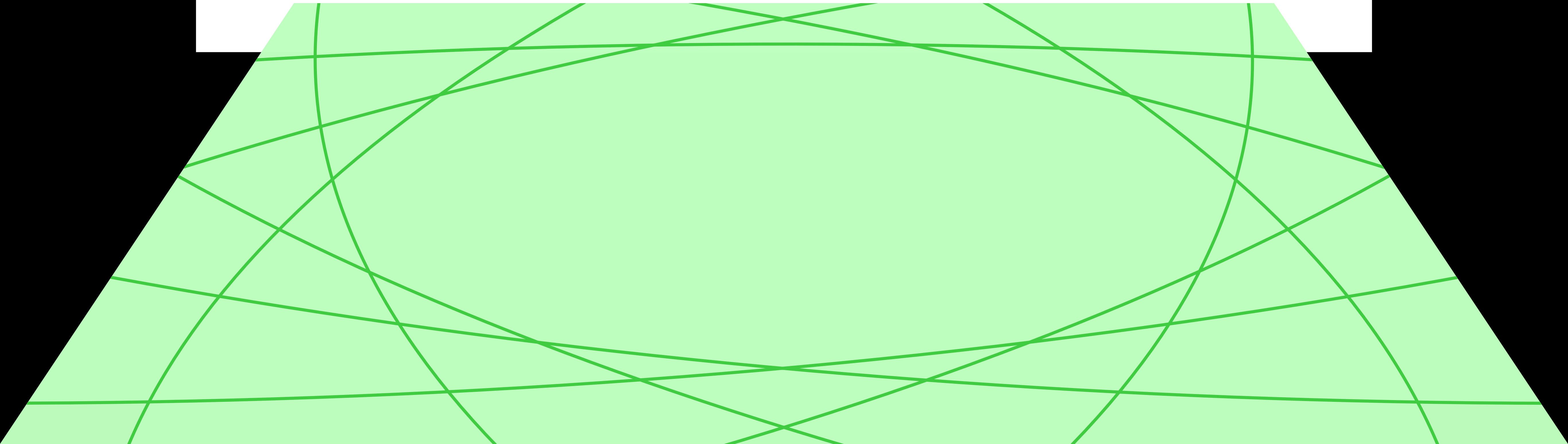
FIRST OBSERVATION...

- Competitive ratio can be pretty bad
  - Let all disks have a common interior
  - Zoom in on the center



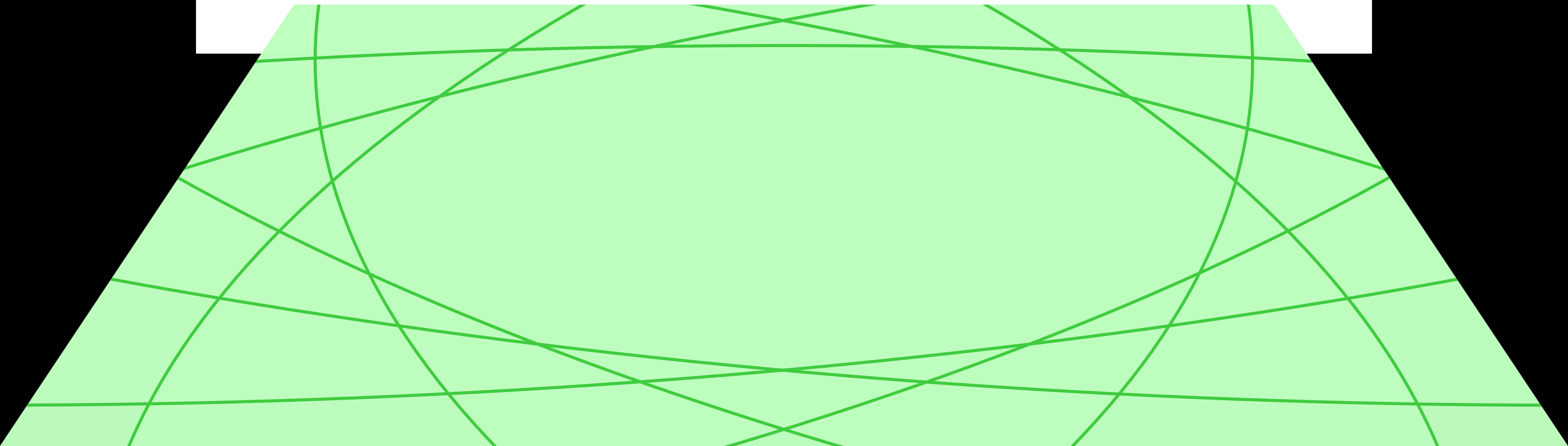
FIRST OBSERVATION...

- Competitive ratio can be pretty bad
  - Let all disks have a common interior
  - Zoom in on the center



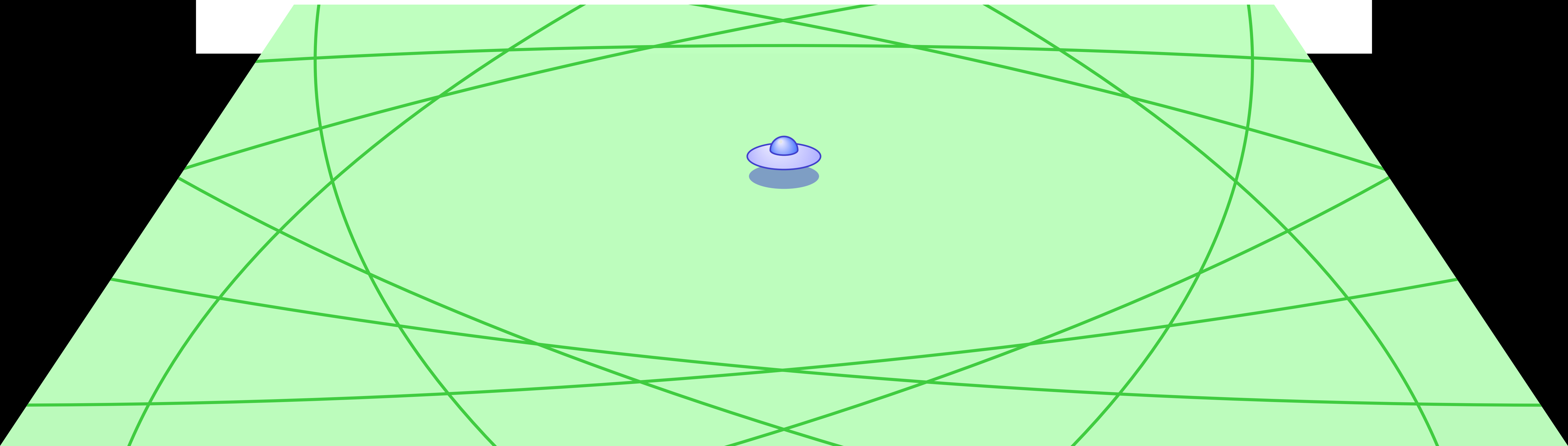
FIRST OBSERVATION...

- Competitive ratio can be pretty bad
  - Let all disks have a common interior
  - Zoom in on the center
- Adversary strategy



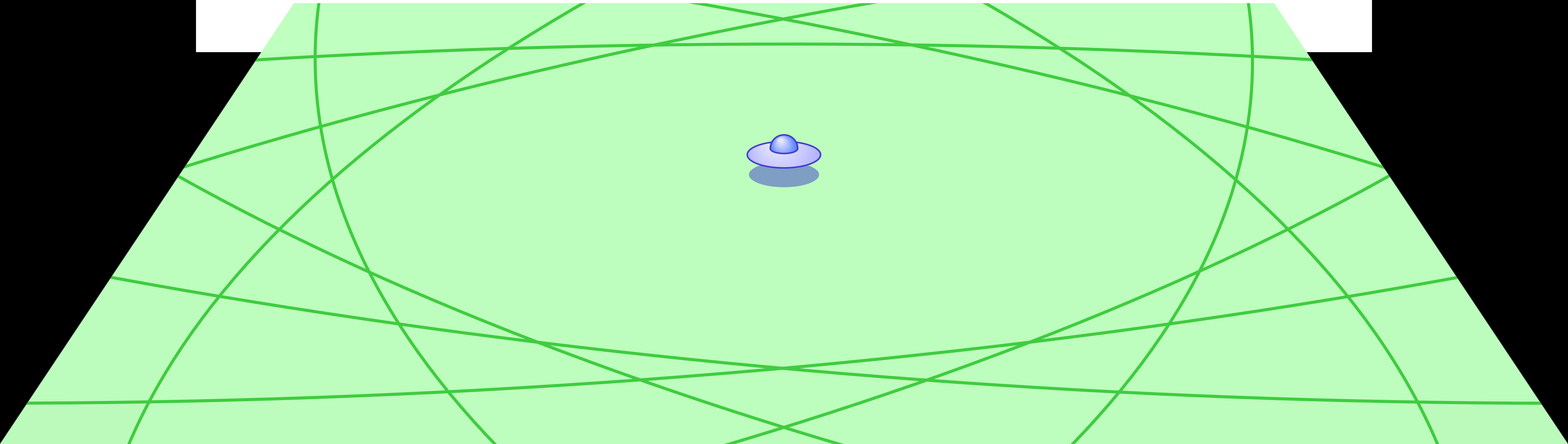
FIRST OBSERVATION...

- Competitive ratio can be pretty bad
  - Let all disks have a common interior
  - Zoom in on the center
- Adversary strategy



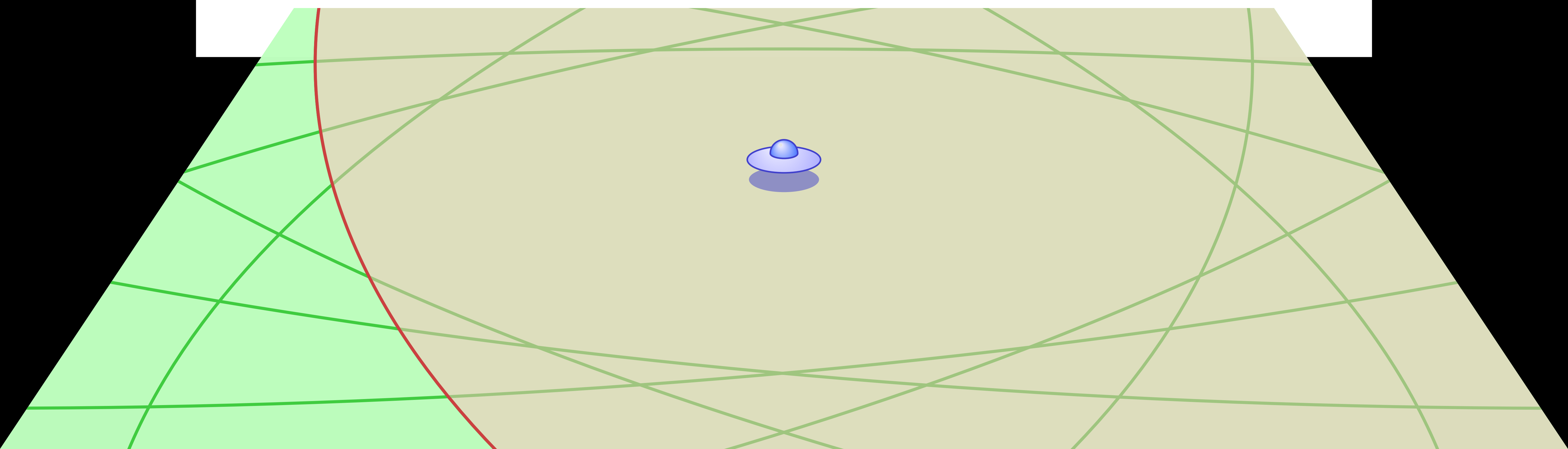
FIRST OBSERVATION...

- Competitive ratio can be pretty bad
  - Let all disks have a common interior
  - Zoom in on the center
- Adversary strategy
  - When algorithm chooses disk i, go to region in every disk except i



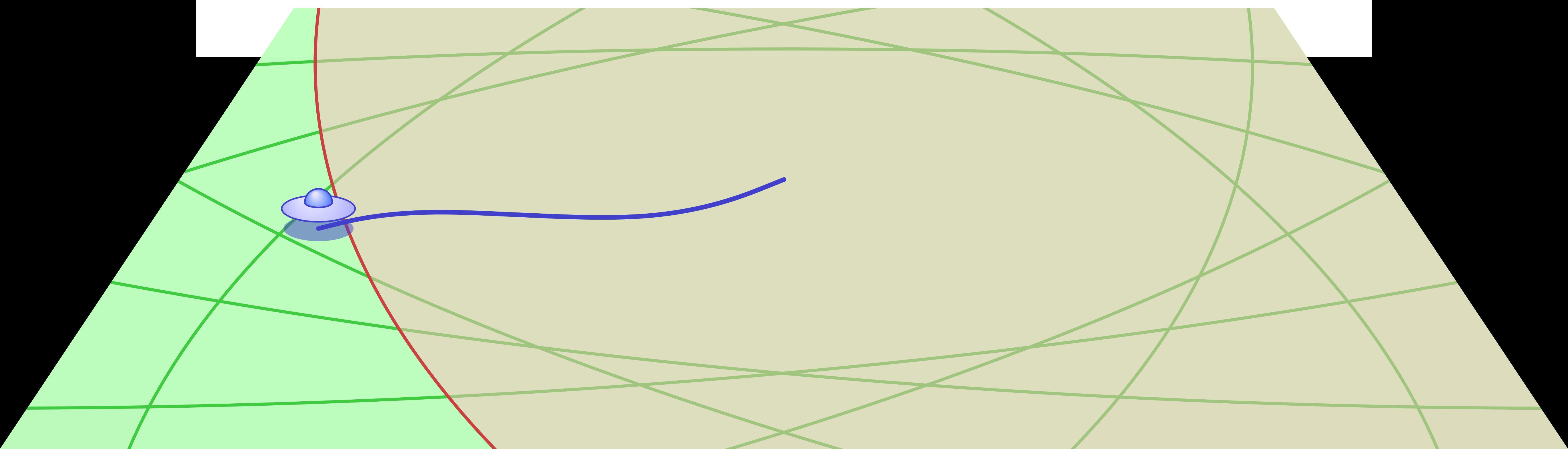
FIRST OBSERVATION...

- Competitive ratio can be pretty bad
  - Let all disks have a common interior
  - Zoom in on the center
- Adversary strategy
  - When algorithm chooses disk i, go to region in every disk except i



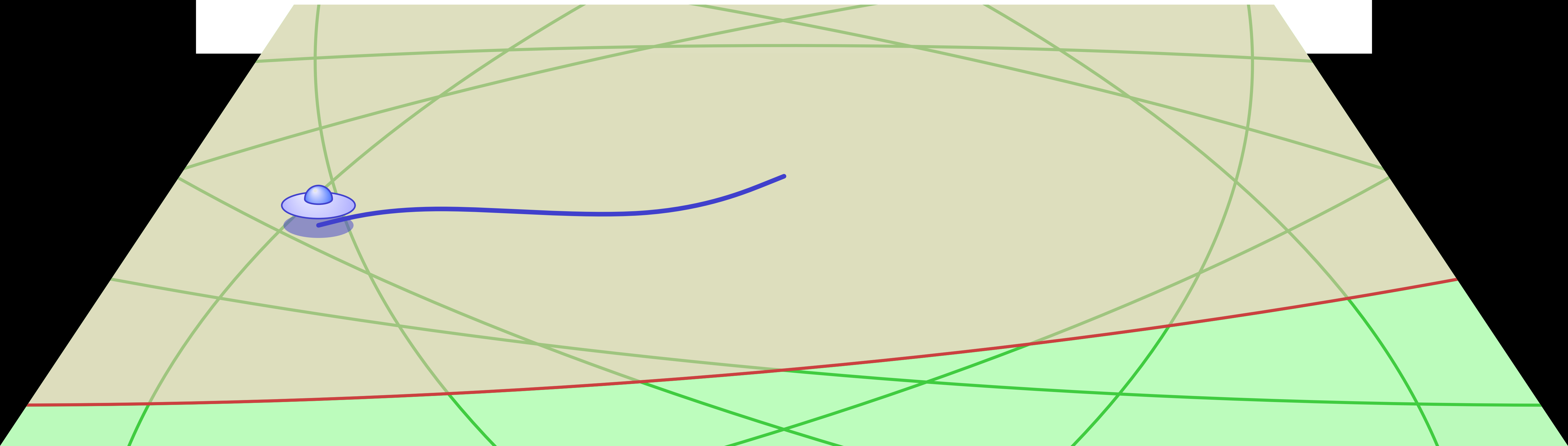
FIRST OBSERVATION...

- Competitive ratio can be pretty bad
  - Let all disks have a common interior
  - Zoom in on the center
- Adversary strategy
  - When algorithm chooses disk i, go to region in every disk except i



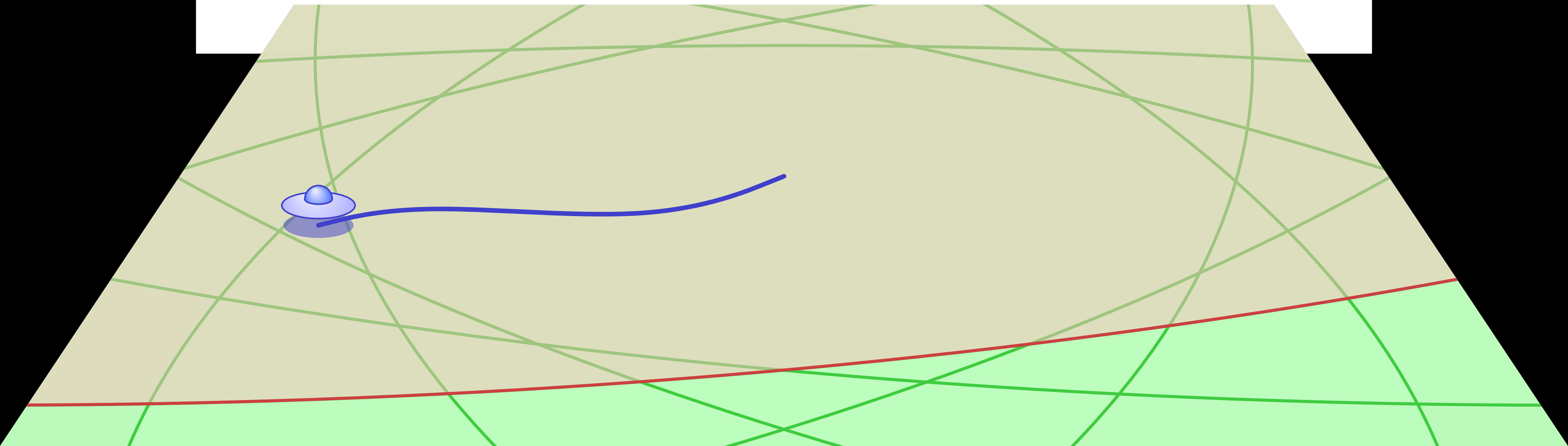
FIRST OBSERVATION…

- Competitive ratio can be pretty bad
  - Let all disks have a common interior
  - Zoom in on the center
- Adversary strategy
  - When algorithm chooses disk i, go to region in every disk except i



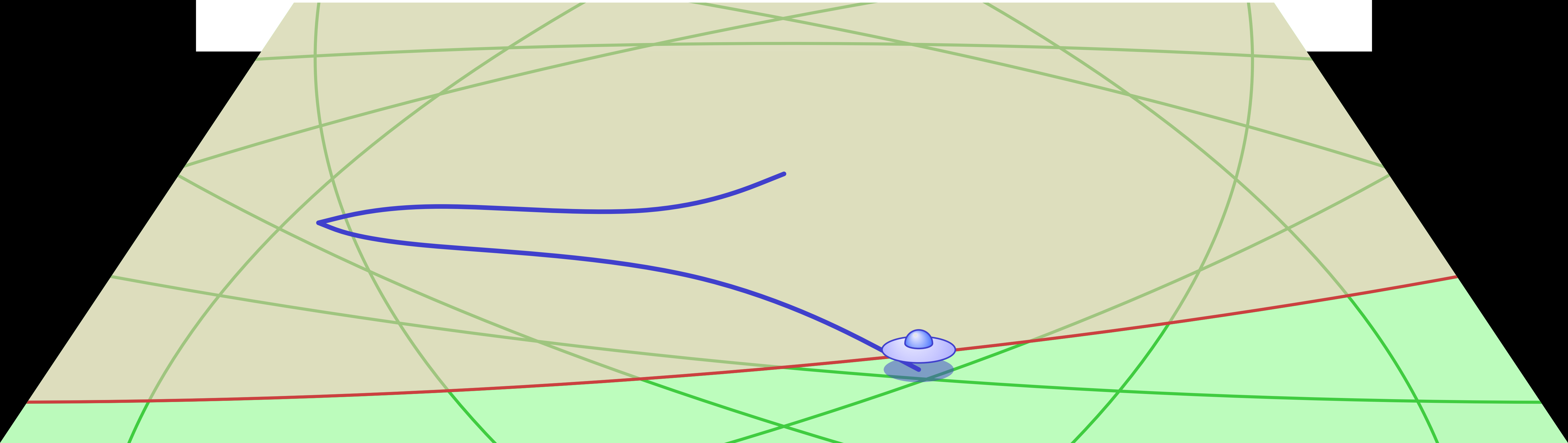
FIRST OBSERVATION...

- Competitive ratio can be pretty bad
  - Let all disks have a common interior
  - Zoom in on the center
- Adversary strategy
  - When algorithm chooses disk i, go to region in every disk except i
  - Repeat $n-1$ times



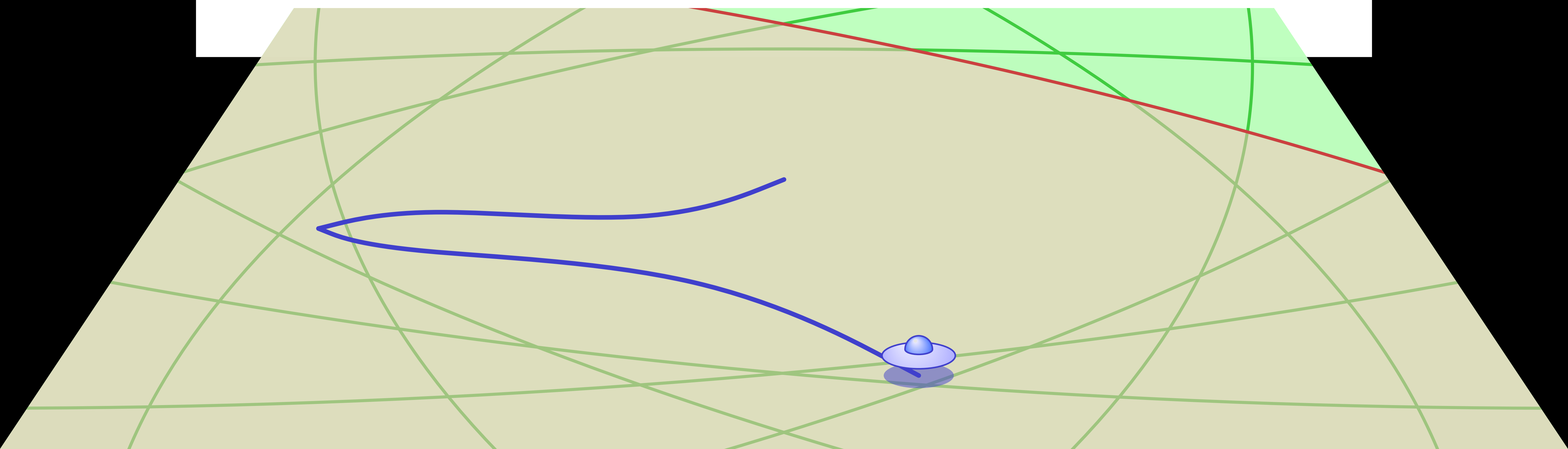
FIRST OBSERVATION...

- Competitive ratio can be pretty bad
  - Let all disks have a common interior
  - Zoom in on the center
- Adversary strategy
  - When algorithm chooses disk i, go to region in every disk except i
  - Repeat $n-1$ times



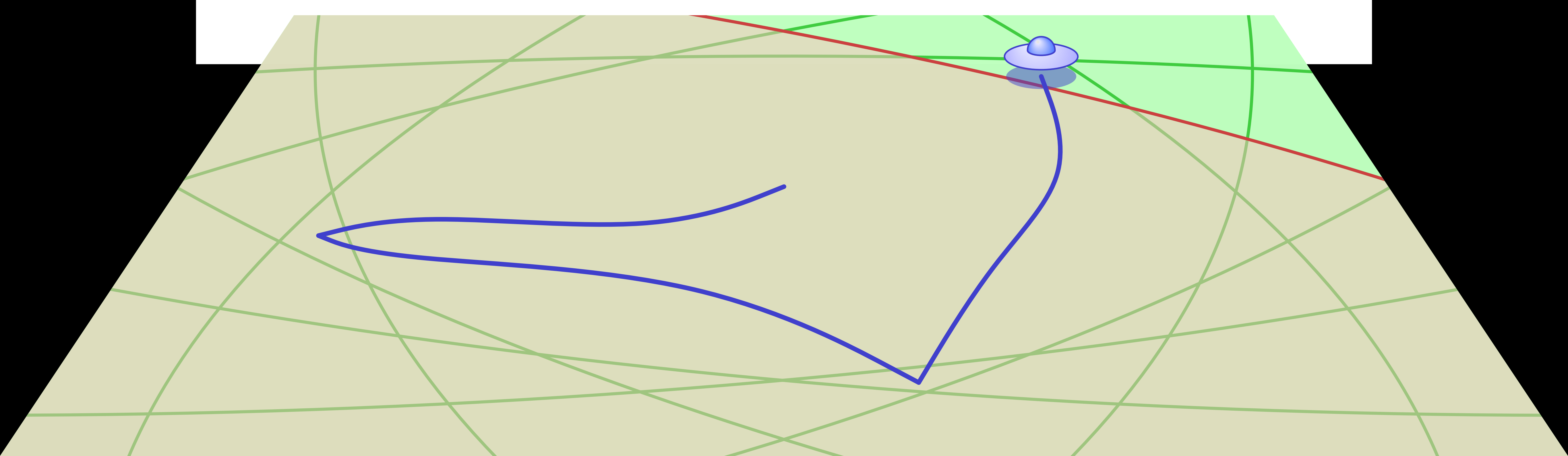
FIRST OBSERVATION...

- Competitive ratio can be pretty bad
  - Let all disks have a common interior
  - Zoom in on the center
- Adversary strategy
  - When algorithm chooses disk i, go to region in every disk except i
  - Repeat $n-1$ times



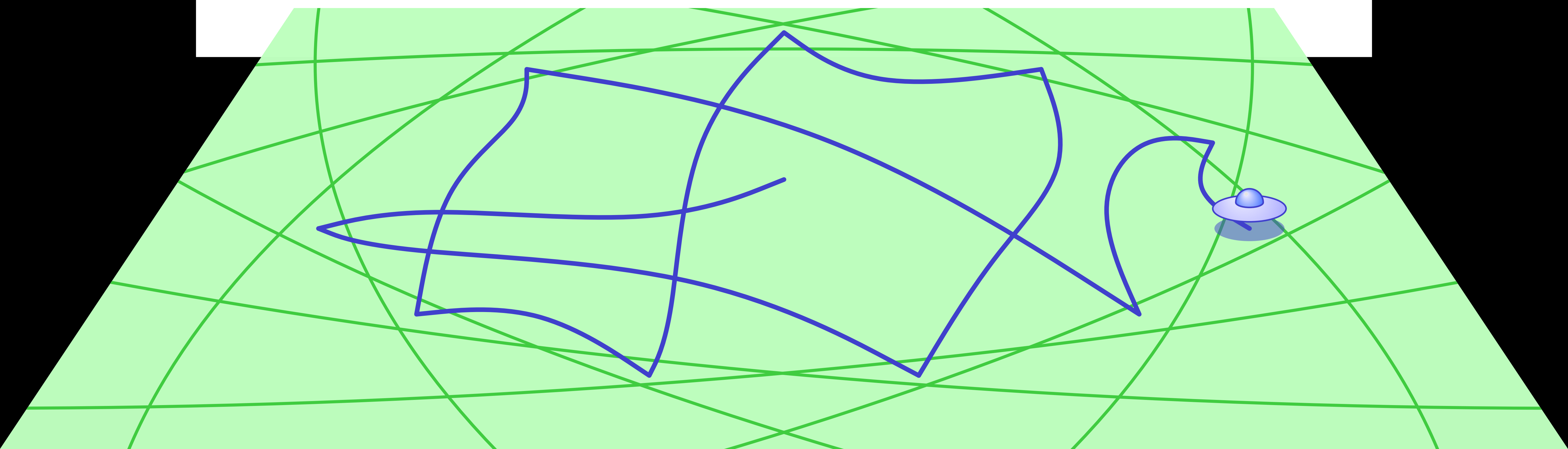
FIRST OBSERVATION...

- Competitive ratio can be pretty bad
  - Let all disks have a common interior
  - Zoom in on the center
- Adversary strategy
  - When algorithm chooses disk i, go to region in every disk except i
  - Repeat $n-1$ times



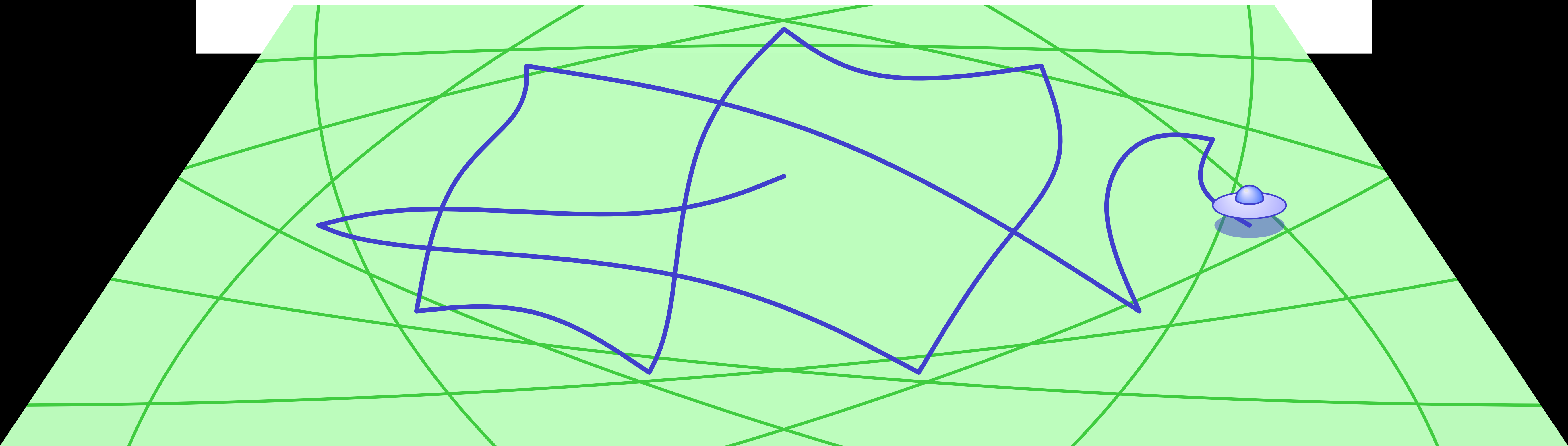
FIRST OBSERVATION...

- Competitive ratio can be pretty bad
  - Let all disks have a common interior
  - Zoom in on the center
- Adversary strategy
  - When algorithm chooses disk i, go to region in every disk except i
  - Repeat $n-1$ times



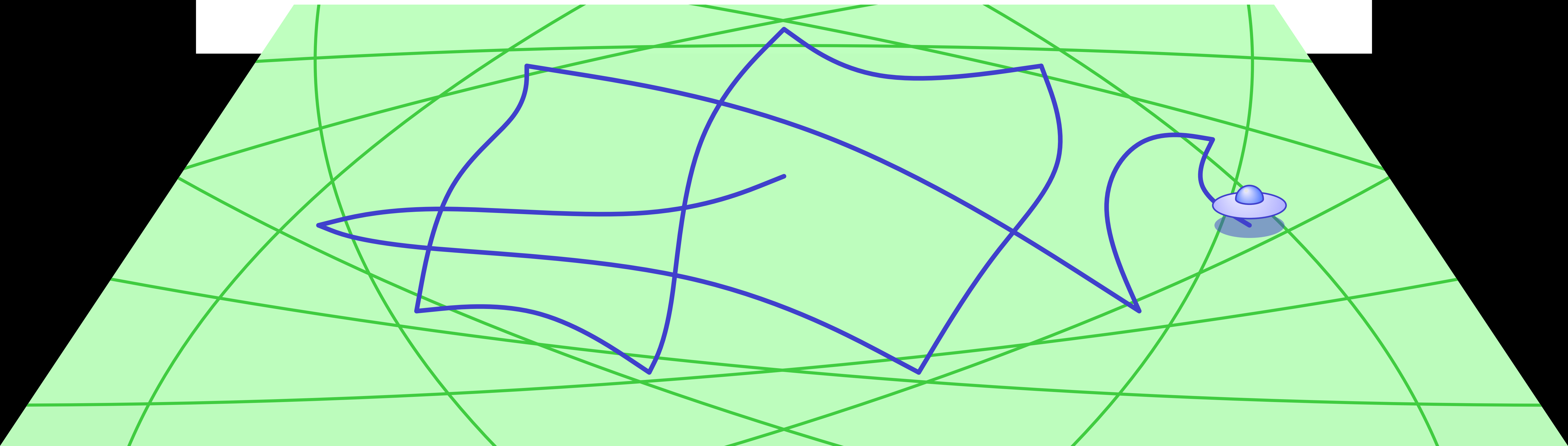
FIRST OBSERVATION...

- Competitive ratio can be pretty bad
  - Let all disks have a common interior
  - Zoom in on the center
- Adversary strategy
  - When algorithm chooses disk i, go to region in every disk except i
  - Repeat $n-1$ times
- Optimal strategy



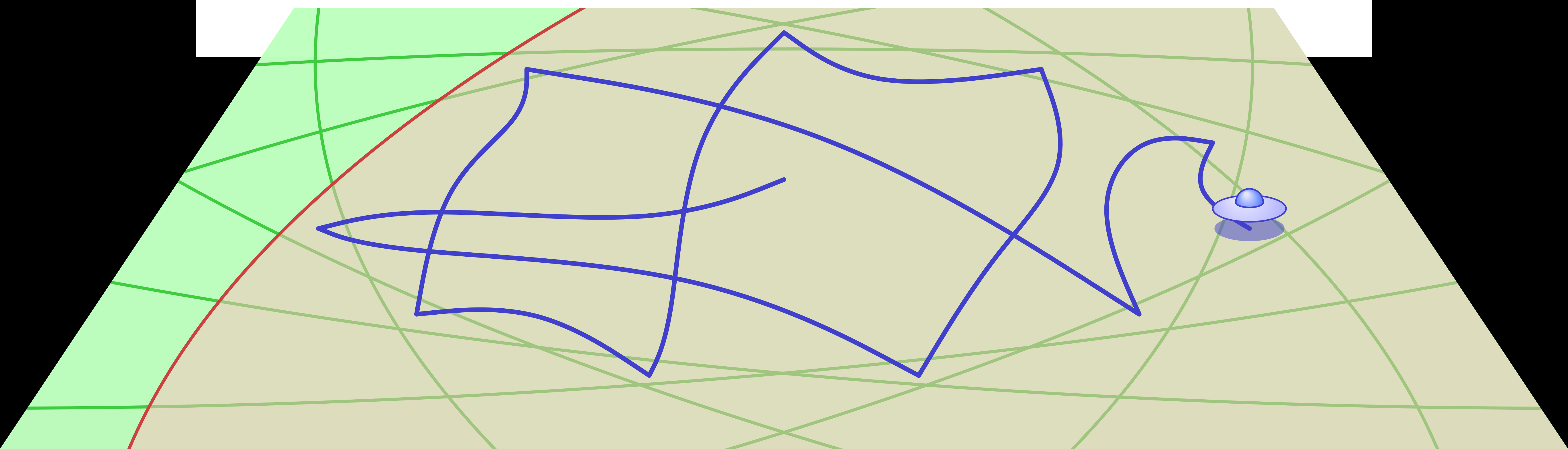
FIRST OBSERVATION...

- Competitive ratio can be pretty bad
  - Let all disks have a common interior
  - Zoom in on the center
- Adversary strategy
  - When algorithm chooses disk i, go to region in every disk except i
  - Repeat $n-1$ times
- Optimal strategy
  - Just take the one remaining disk



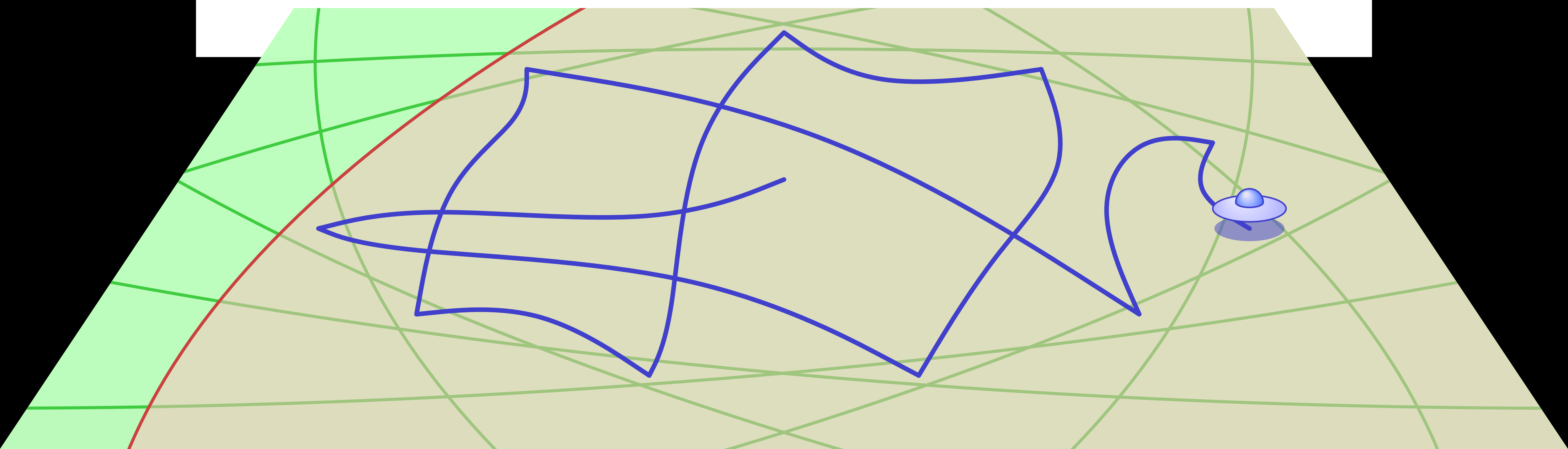
FIRST OBSERVATION...

- Competitive ratio can be pretty bad
  - Let all disks have a common interior
  - Zoom in on the center
- Adversary strategy
  - When algorithm chooses disk i, go to region in every disk except i
  - Repeat $n-1$ times
- Optimal strategy
  - Just take the one remaining disk



FIRST OBSERVATION...

- Competitive ratio can be pretty bad
  - Let all disks have a common interior
  - Zoom in on the center
- Adversary strategy
  - When algorithm chooses disk i, go to region in every disk except i
  - Repeat $n - 1$ times
- Optimal strategy
  - Just take the one remaining disk
  - Competitive ratio is $n - 1$



PLY

PLY

- Lower bound requires many overlapping disks

PLY

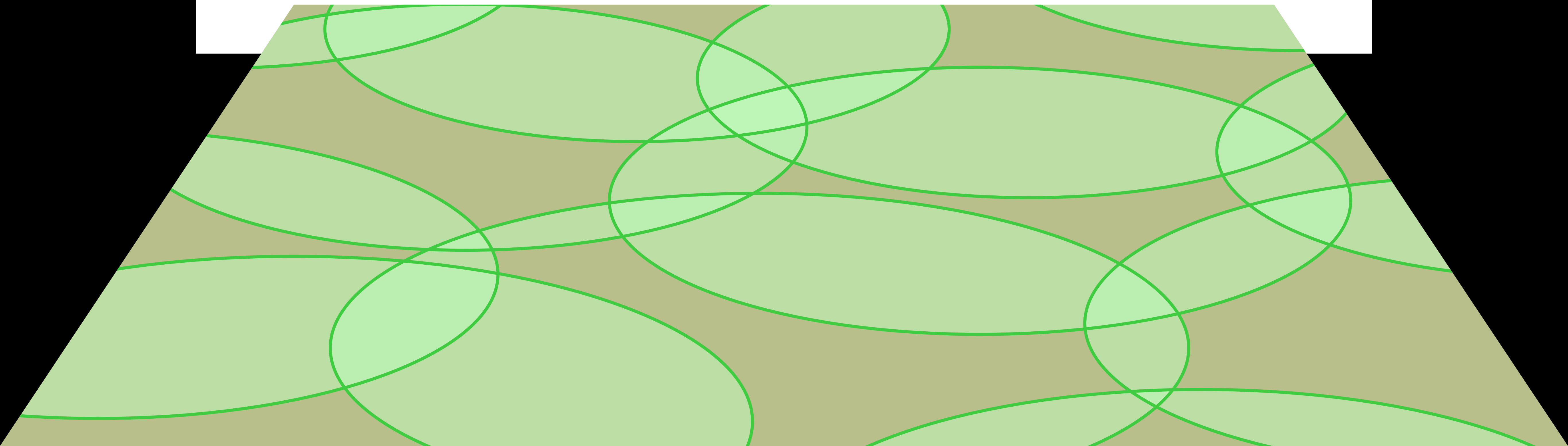
- Lower bound requires many overlapping disks
- Ply of a point

PLY

- Lower bound requires many overlapping disks
- Ply of a point
  - Given a set of regions

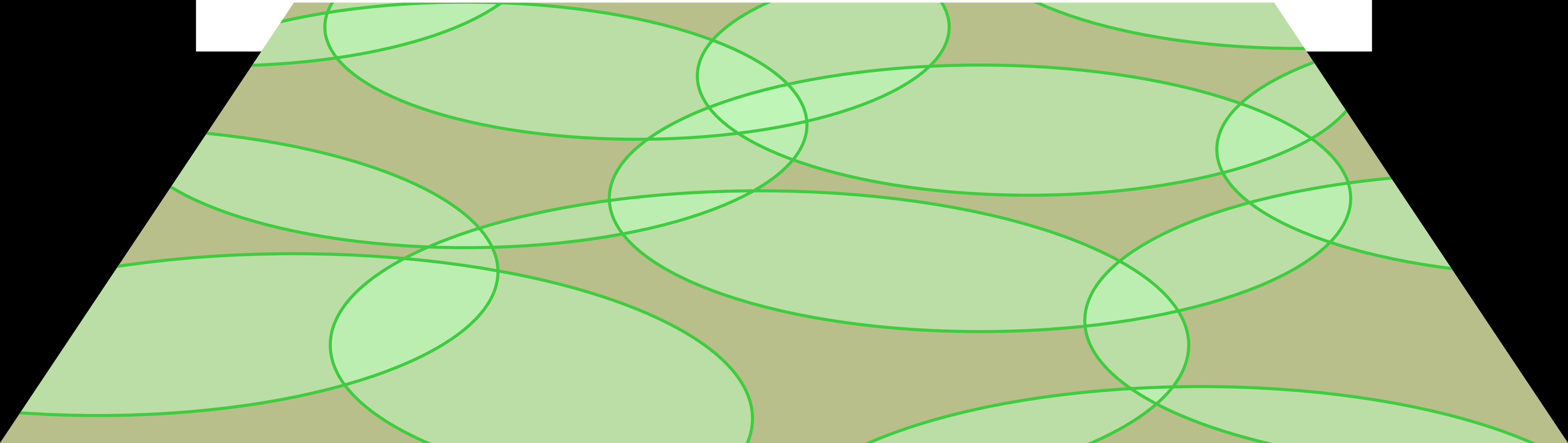
PLY

- Lower bound requires many overlapping disks
- Ply of a point
  - Given a set of regions



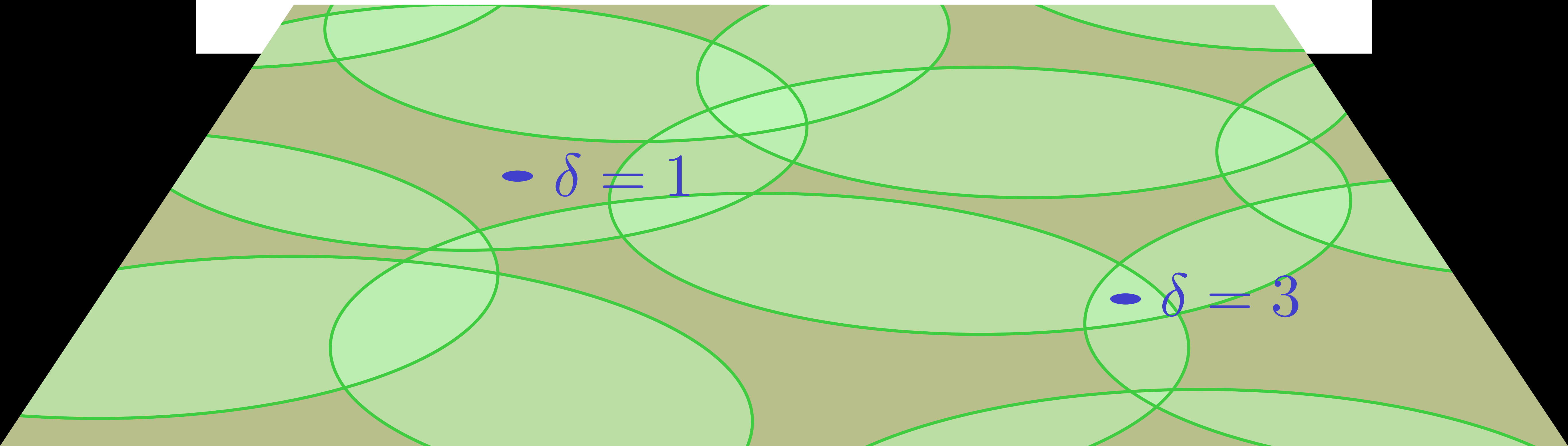
PLY

- Lower bound requires many overlapping disks
- Ply of a point
  - Given a set of regions
  - $\delta(p) = |\{R : p \in R\}|$



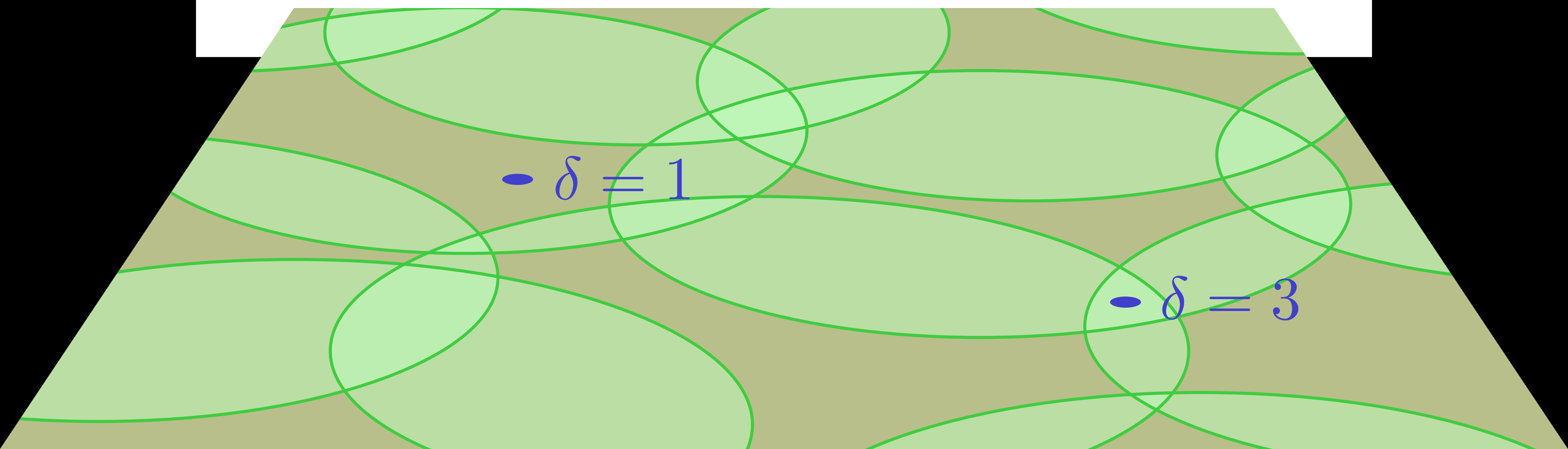
PLY

- Lower bound requires many overlapping disks
- Ply of a point
  - Given a set of regions
  - $\delta(p) = |\{R : p \in R\}|$



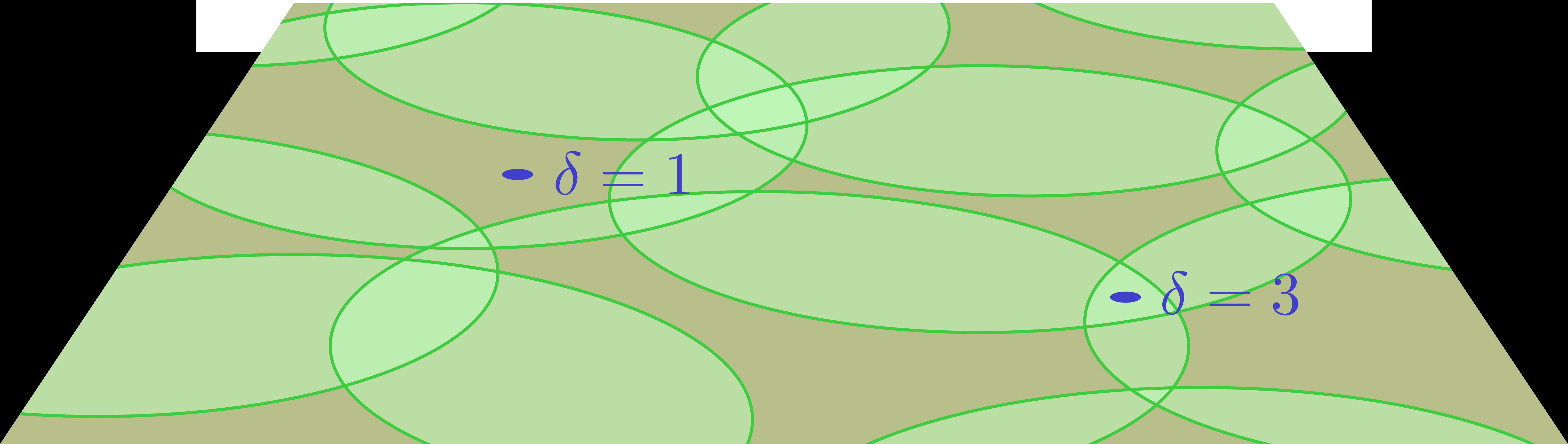
PLY

- Lower bound requires many overlapping disks
- Ply of a point
  - Given a set of regions
  - $\delta(p) = |\{R : p \in R\}|$
- Ply of the set of regions



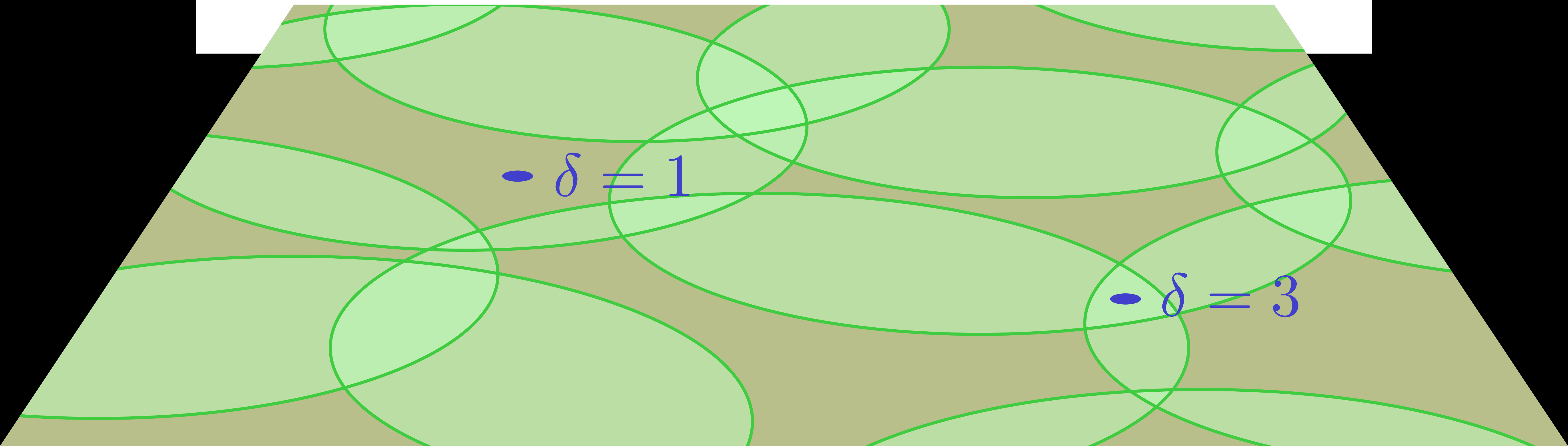
PLY

- Lower bound requires many overlapping disks
- Ply of a point
  - Given a set of regions
  - $\delta(p) = |\{R : p \in R\}|$
- Ply of the set of regions
  - Maximum ply of all points



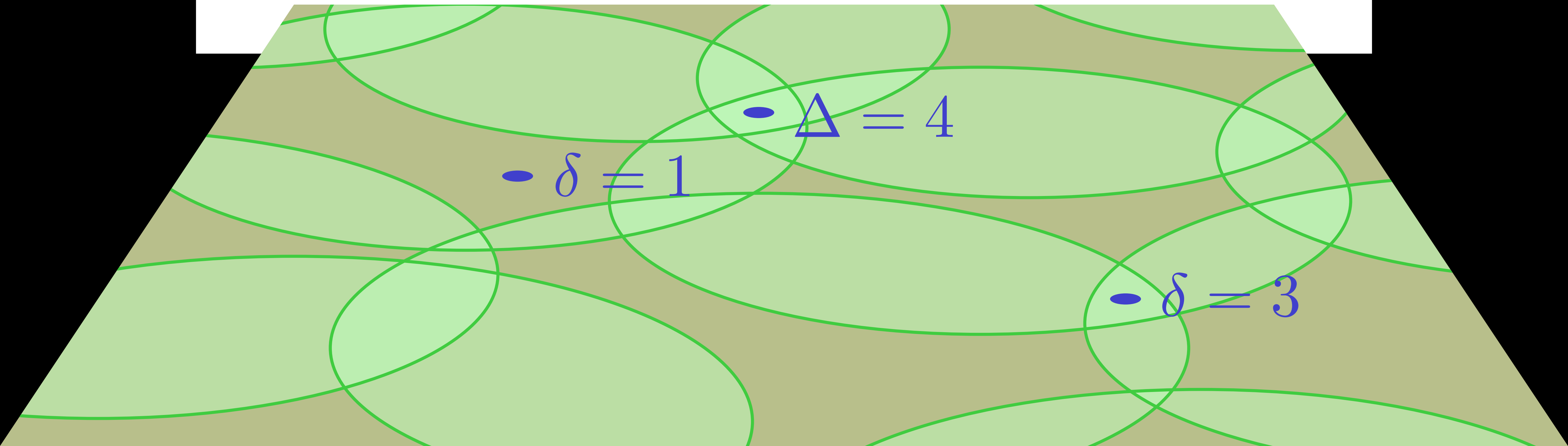
PLY

- Lower bound requires many overlapping disks
- Ply of a point
  - Given a set of regions
  - $\delta(p) = |\{R : p \in R\}|$
- Ply of the set of regions
  - Maximum ply of all points
  - $\Delta = \max_p \delta(p)$



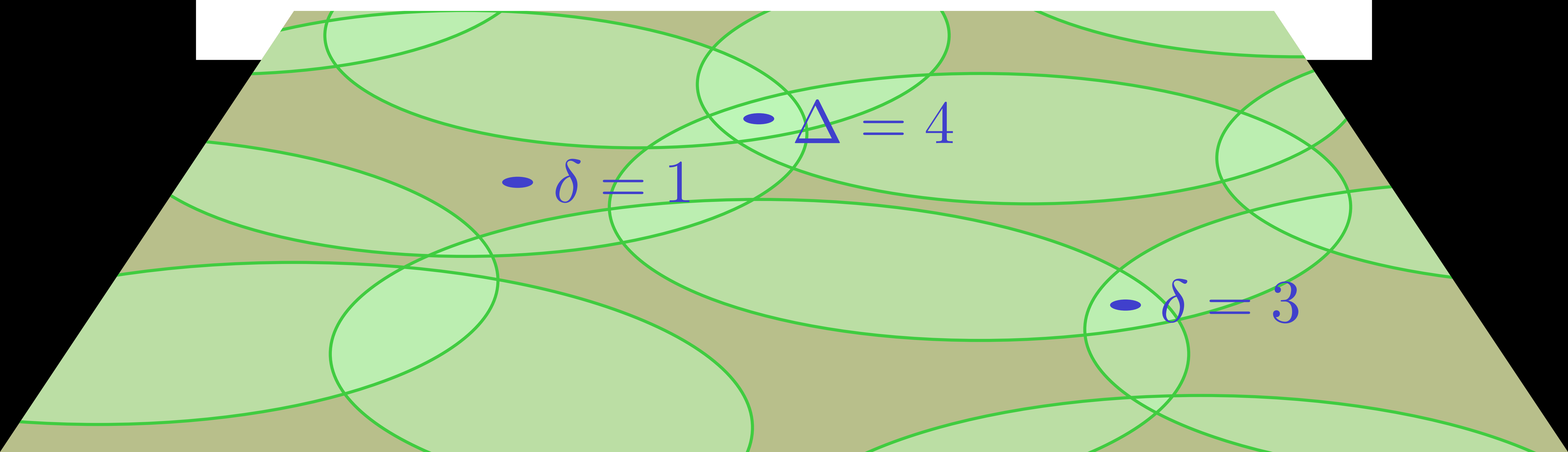
PLY

- Lower bound requires many overlapping disks
- Ply of a point
  - Given a set of regions
  - $\delta(p) = |\{R : p \in R\}|$
- Ply of the set of regions
  - Maximum ply of all points
  - $\Delta = \max_p \delta(p)$



PLY

- Lower bound requires many overlapping disks
- Ply of a point
  - Given a set of regions
  - $\delta(p) = |\{R : p \in R\}|$
- Ply of the set of regions
  - Maximum ply of all points
  - $\Delta = \max_p \delta(p)$
- In practice, $\Delta \ll n$



RESULTS

RESULTS

- Bounds on competitive ratio

RESULTS

- Bounds on competitive ratio
  - Stateless unbounded

RESULTS

- Bounds on competitive ratio
  - Stateless unbounded
  - Deterministic $(d = 1)$ $\Theta(\log(\Delta - c))$

RESULTS

- Bounds on competitive ratio
  - Stateless — unbounded
  - Deterministic $(d = 1)$ — $\Theta(\log(\Delta - c))$
    $(d > 1)$ — $\Theta(\Delta - c)$

RESULTS

- Bounds on competitive ratio
  - Stateless — unbounded
  - Deterministic $(d = 1)$ — $\Theta(\log(\Delta - c))$
    $(d > 1)$ — $\Theta(\Delta - c)$
  - Randomised — $\Theta(\log(\Delta - c))$

RESULTS

- Bounds on competitive ratio
  - Stateless — unbounded
  - Deterministic $(d = 1)$ — $\Theta(\log(\Delta - c))$
    $(d > 1)$ — $\Theta(\Delta - c)$
  - Randomised — $\Theta(\log(\Delta - c))$
- Running time

RESULTS

- Bounds on competitive ratio
  - Stateless — unbounded
  - Deterministic $(d = 1)$ — $\Theta(\log(\Delta - c))$
    $(d > 1)$ — $\Theta(\Delta - c)$
  - Randomised — $\Theta(\log(\Delta - c))$
- Running time
  - Assume input is given as event sequence

RESULTS

- Bounds on competitive ratio
  - Stateless — unbounded
  - Deterministic $(d = 1)$ — $\Theta(\log(\Delta - c))$
    $(d > 1)$ — $\Theta(\Delta - c)$
  - Randomised — $\Theta(\log(\Delta - c))$
- Running time
  - Assume input is given as event sequence
  - Constant amortised time per event

OFFLINE

OFFLINE

- Definitions

OFFLINE

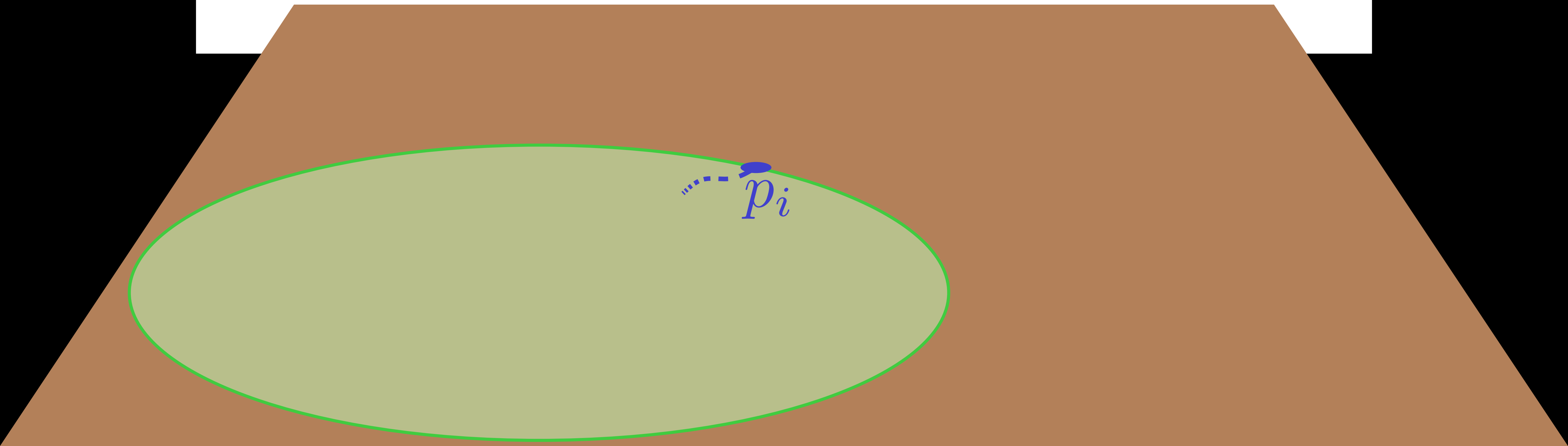
- Definitions

OFFLINE

- Definitions
  - p_i: point where a new station is necessary

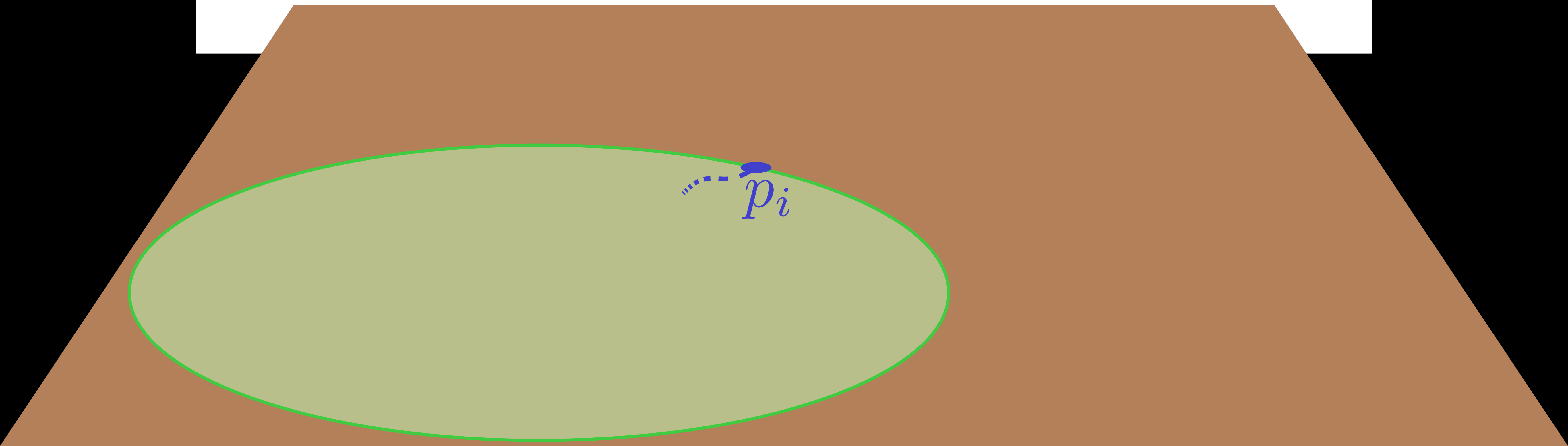
OFFLINE

- Definitions
  - p_i: point where a new station is necessary



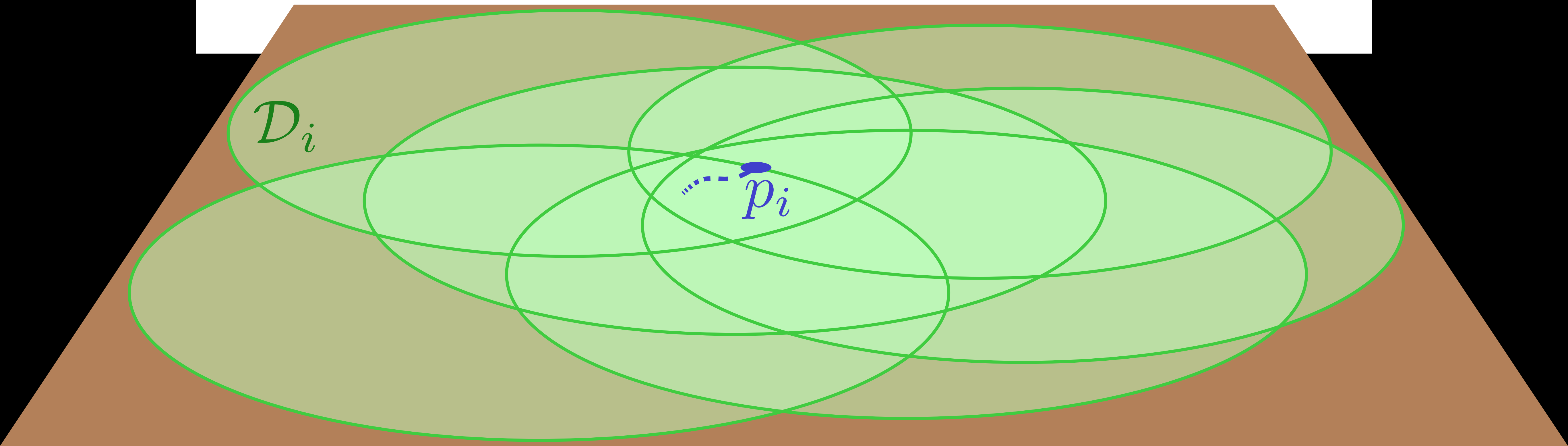
OFFLINE

- Definitions
  - p_i: point where a new station is necessary
  - $\mathcal{D}_i$: set of regions that contain p_i



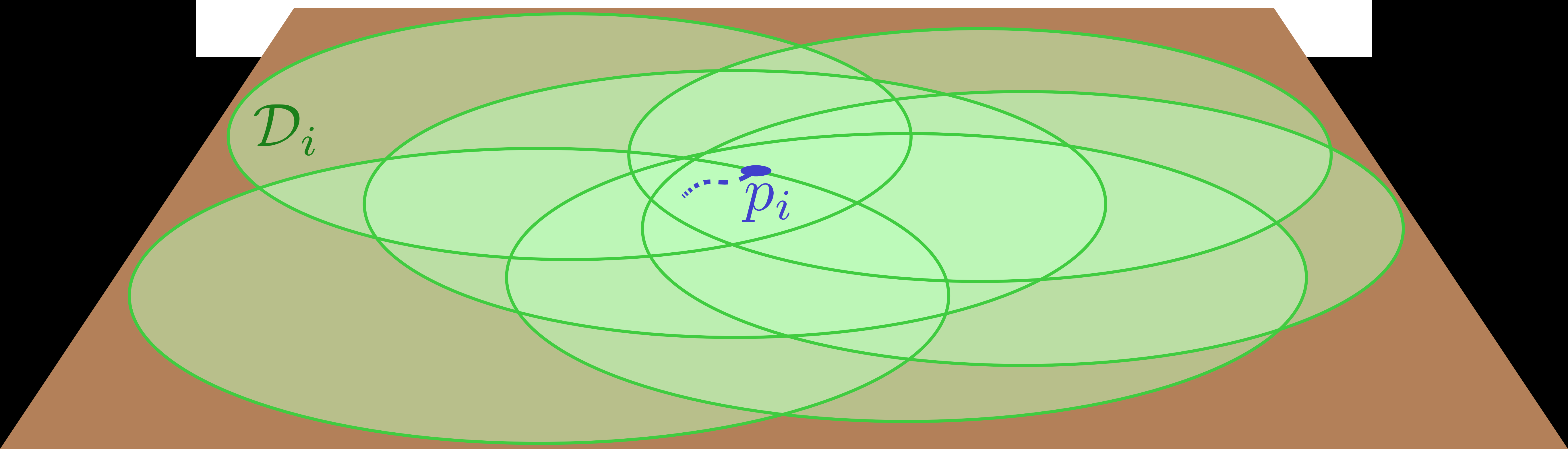
OFFLINE

- Definitions
  - p_i: point where a new station is necessary
  - $\mathcal{D}_i$: set of regions that contain p_i



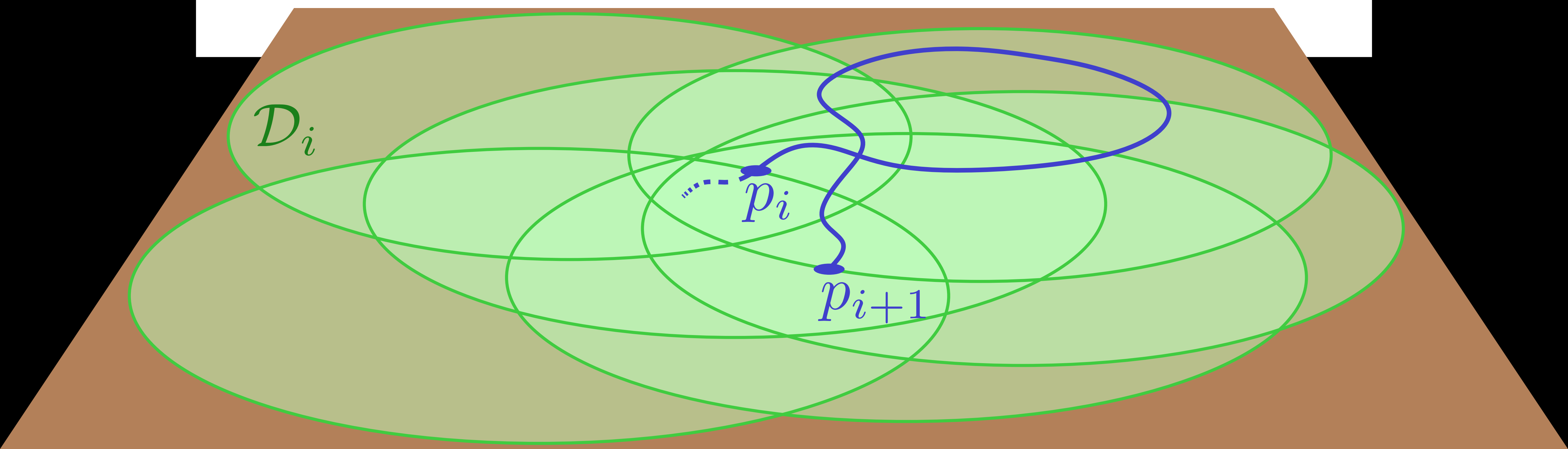
OFFLINE

- Definitions
  - p_i: point where a new station is necessary
  - $\mathcal{D}_i$: set of regions that contain p_i
  - p_{i+1}: point where the object leaves the last disk of $\mathcal{D}_i$



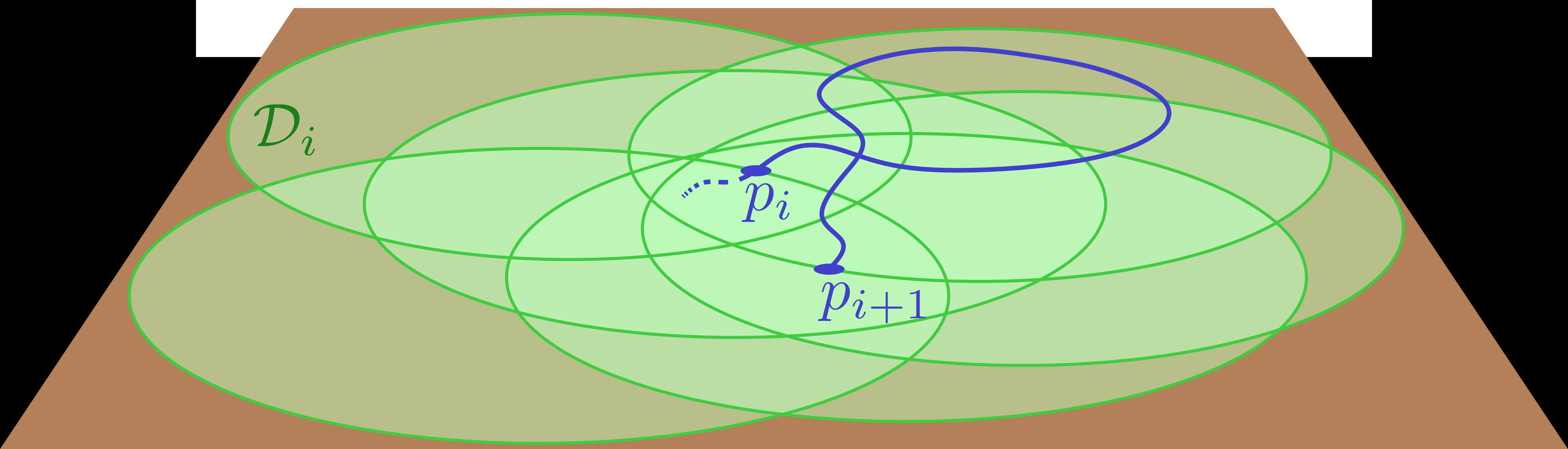
OFFLINE

- Definitions
  - p_i: point where a new station is necessary
  - $\mathcal{D}_i$: set of regions that contain p_i
  - p_{i+1}: point where the object leaves the last disk of $\mathcal{D}_i$



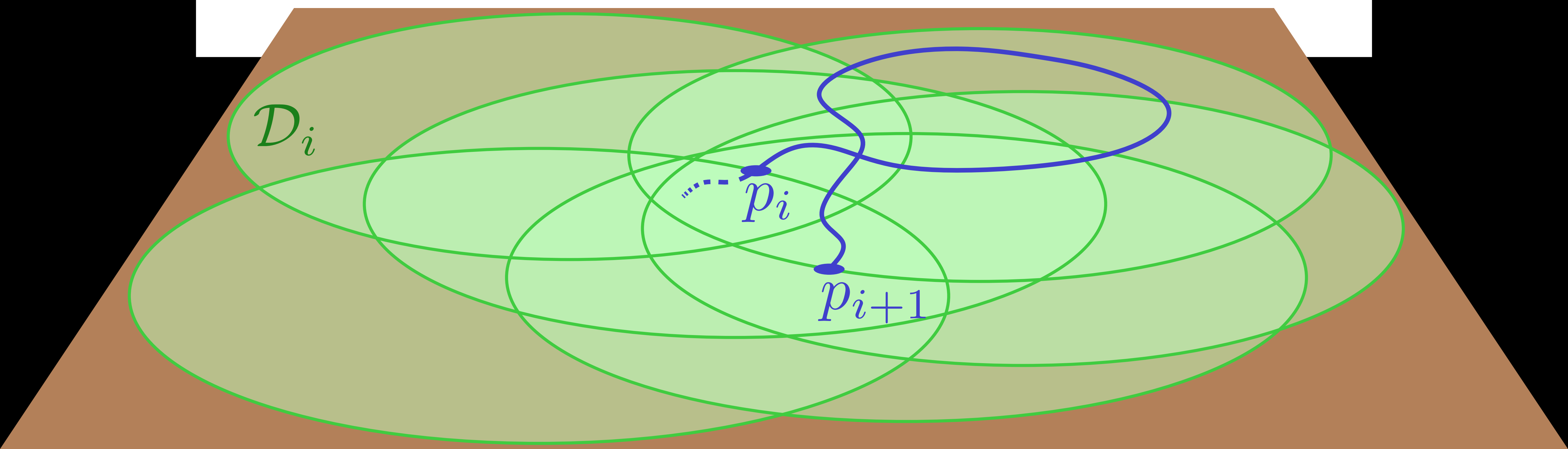
OFFLINE

- Definitions
  - p_i: point where a new station is necessary
  - $\mathcal{D}_i$: set of regions that contain p_i
  - p_{i+1}: point where the object leaves the last disk of $\mathcal{D}_i$
- Optimal solution



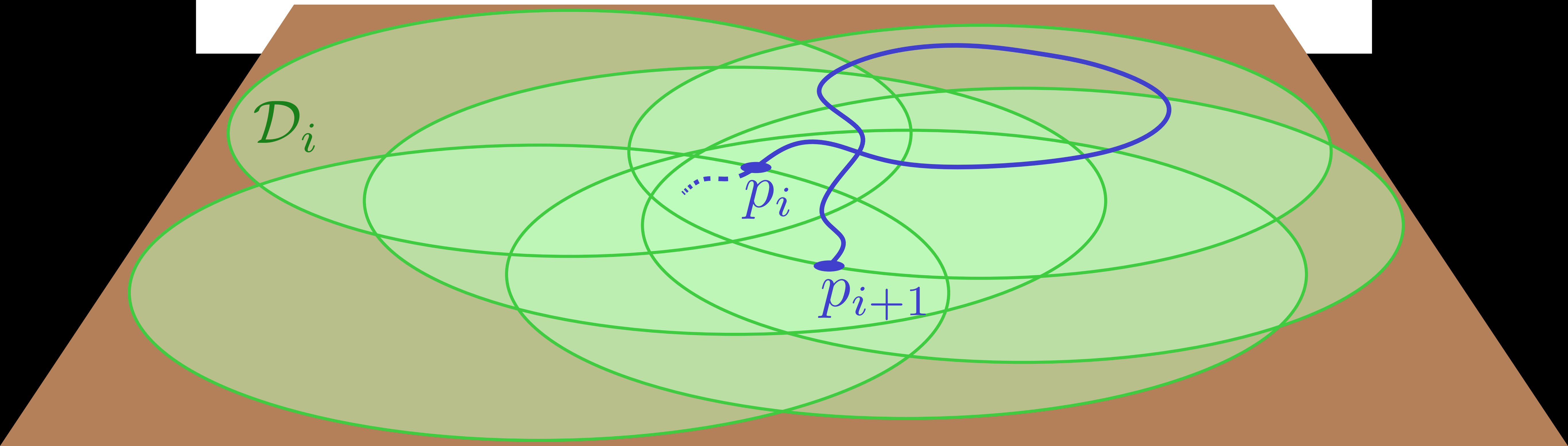
OFFLINE

- Definitions
  - p_i: point where a new station is necessary
  - $\mathcal{D}_i$: set of regions that contain p_i
  - p_{i+1}: point where the object leaves the last disk of $\mathcal{D}_i$
- Optimal solution
  - Set p_0 to be the start point

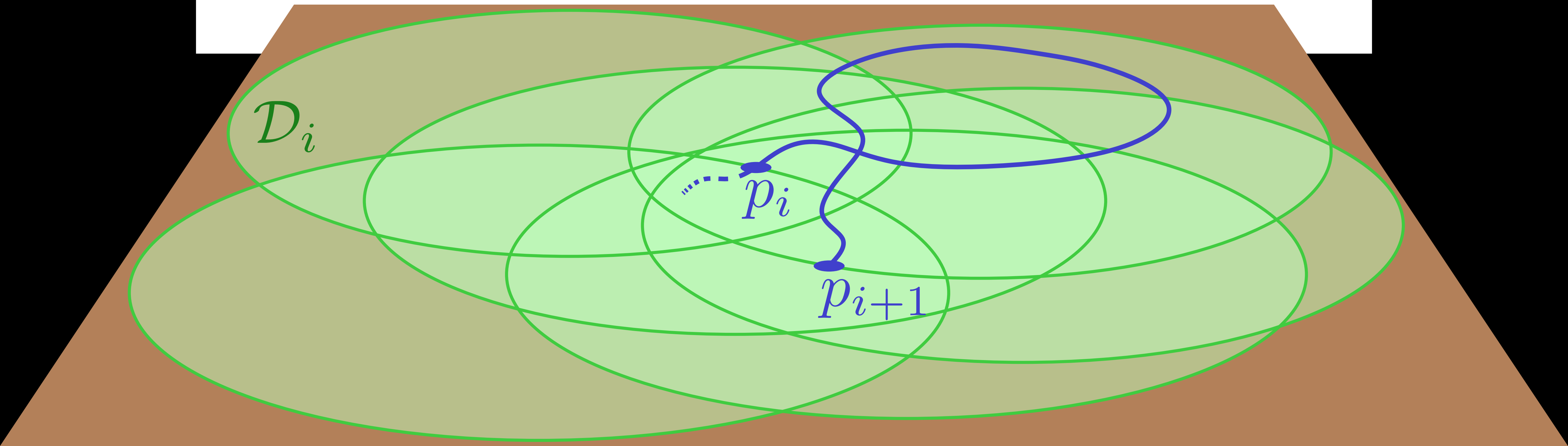


OFFLINE

- Definitions
  - p_i: point where a new station is necessary
  - $\mathcal{D}_i$: set of regions that contain p_i
  - p_{i+1}: point where the object leaves the last disk of $\mathcal{D}_i$
- Optimal solution
  - Set p_0 to be the start point
  - Optimal cost up to p_k is k

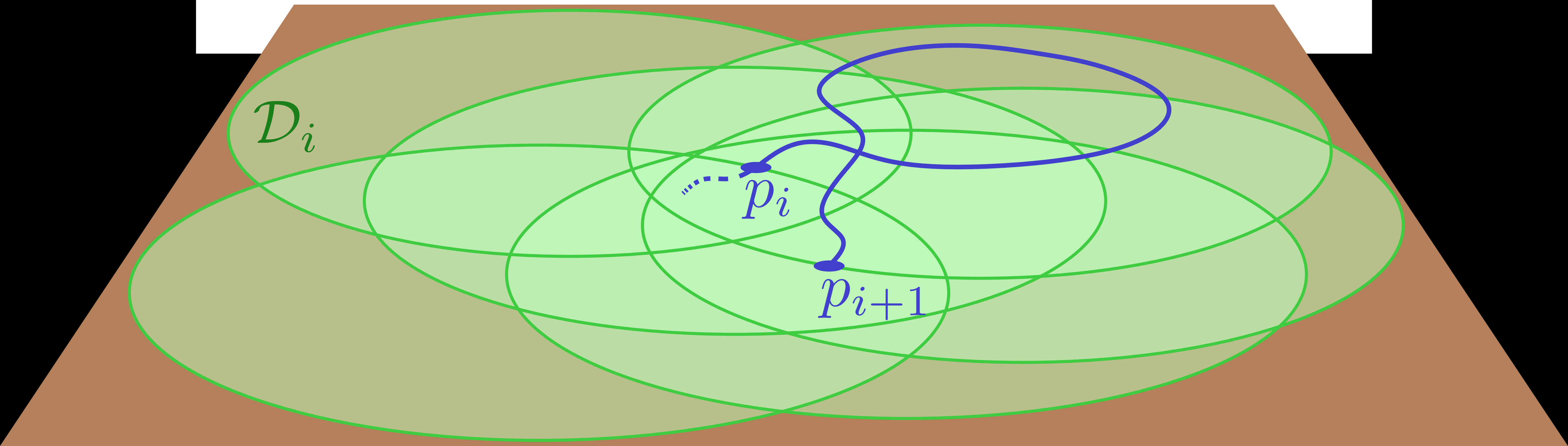


DETERMINISTIC ONLINE



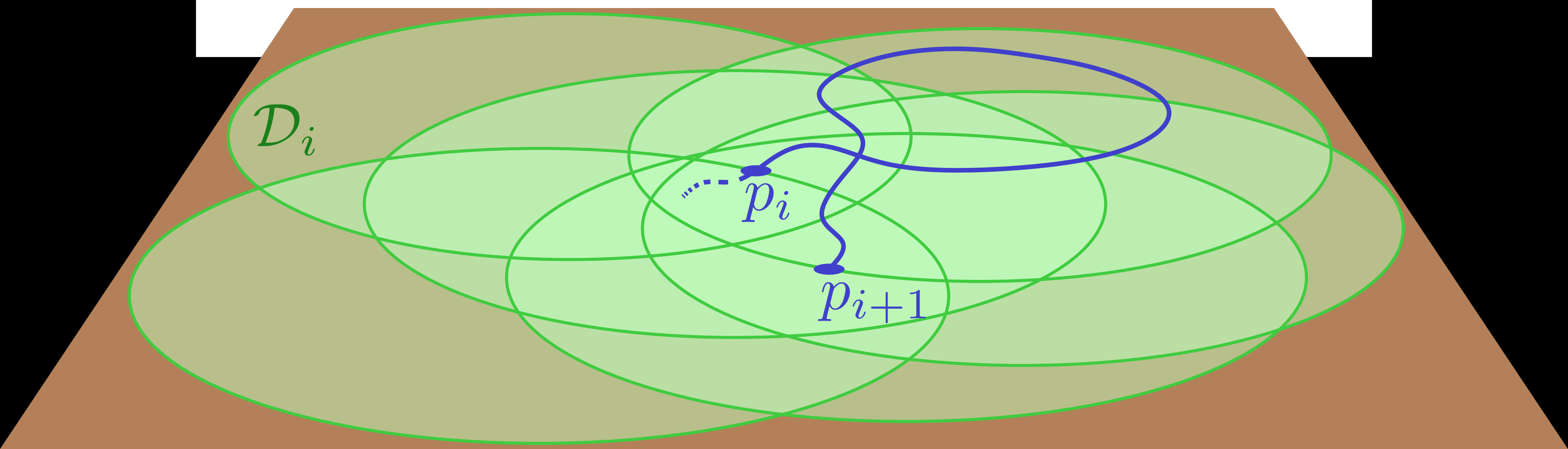
DETERMINISTIC ONLINE

- Algorithm



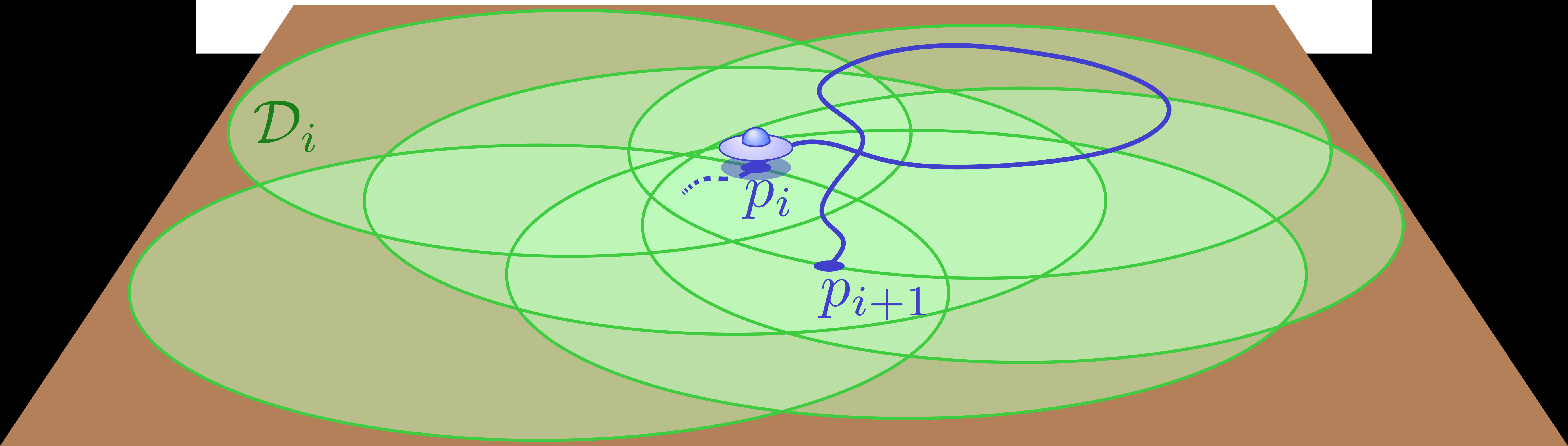
DETERMINISTIC ONLINE

- Algorithm
  - When at p_i, compute $\mathcal{D}_i$



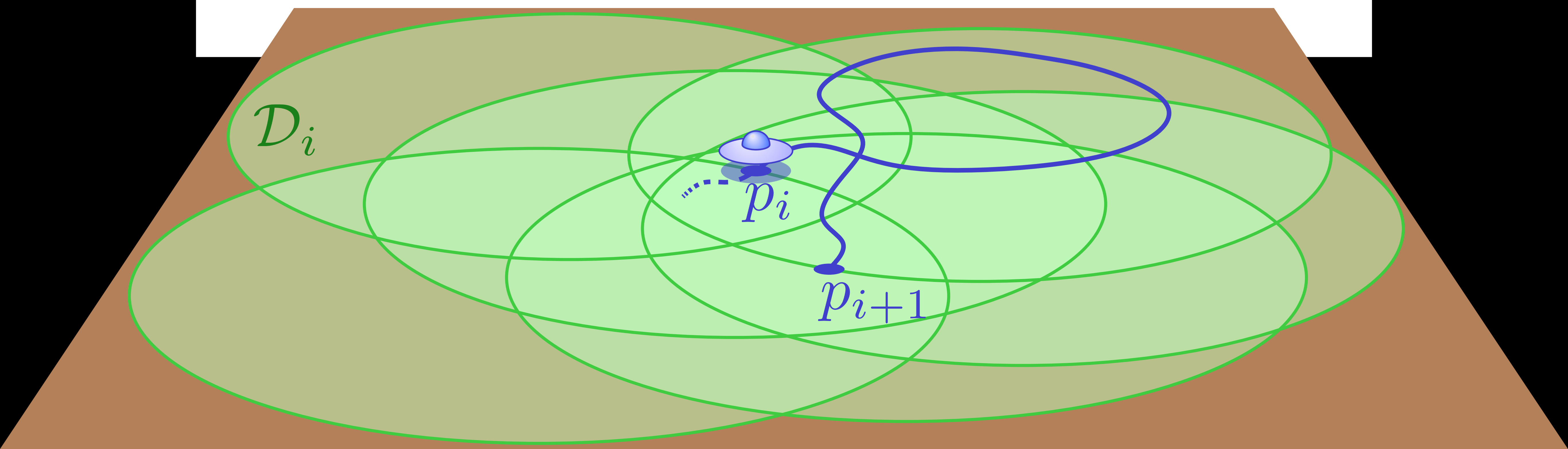
DETERMINISTIC ONLINE

- Algorithm
  - When at p_i, compute $\mathcal{D}_i$



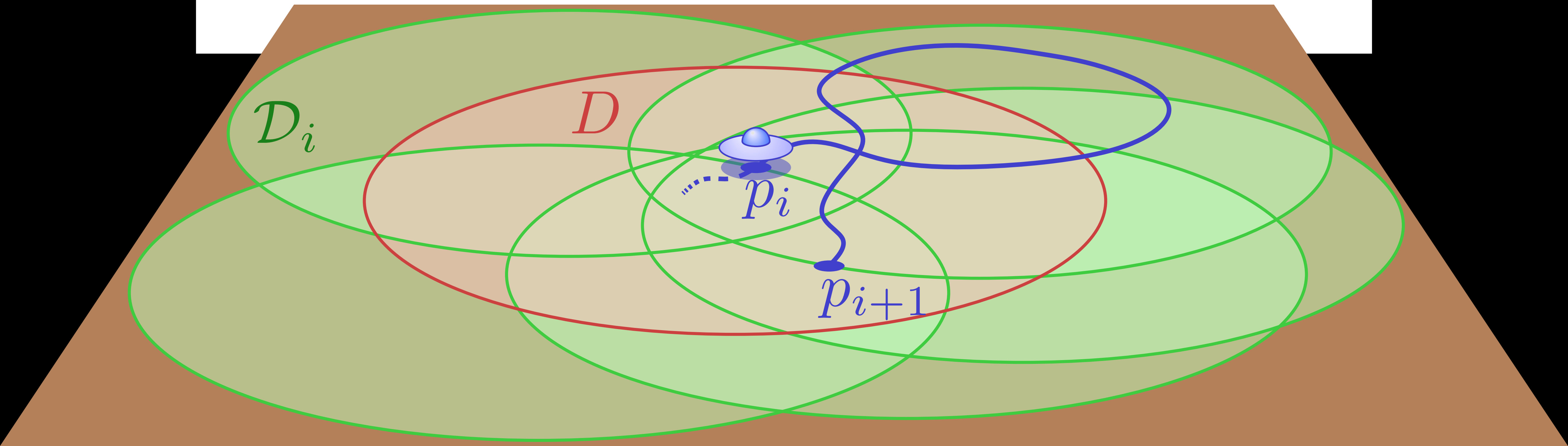
DETERMINISTIC ONLINE

- Algorithm
  - When at p_i, compute $\mathcal{D}_i$
  - Let D be "first" region of $\mathcal{D}_i$



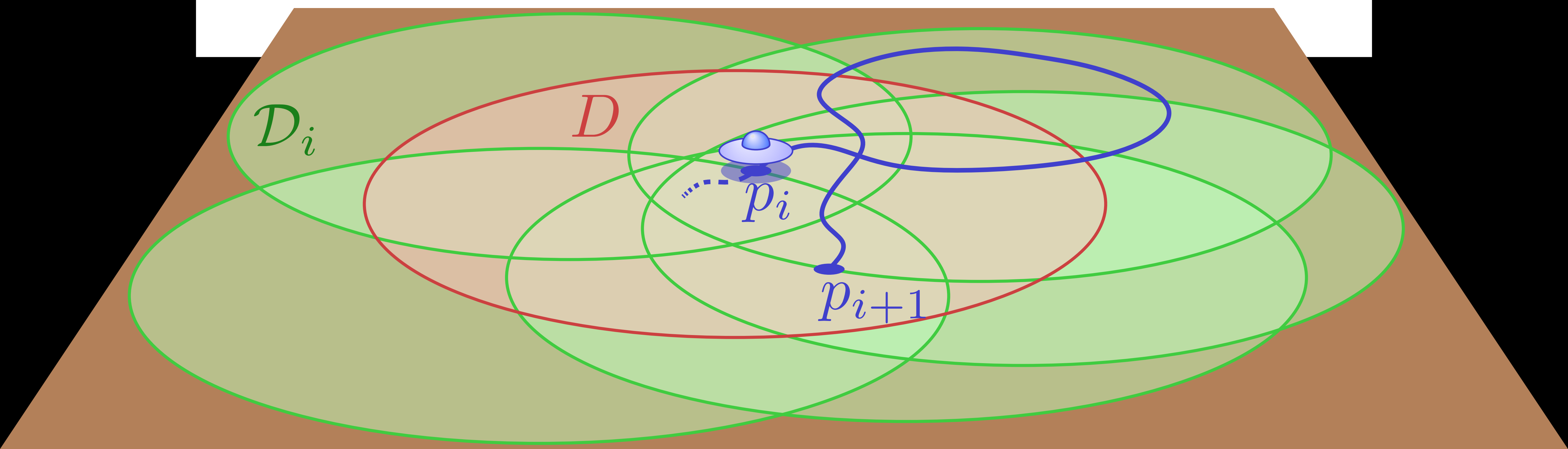
DETERMINISTIC ONLINE

- Algorithm
  - When at p_i, compute $\mathcal{D}_i$
  - Let D be "first" region of $\mathcal{D}_i$



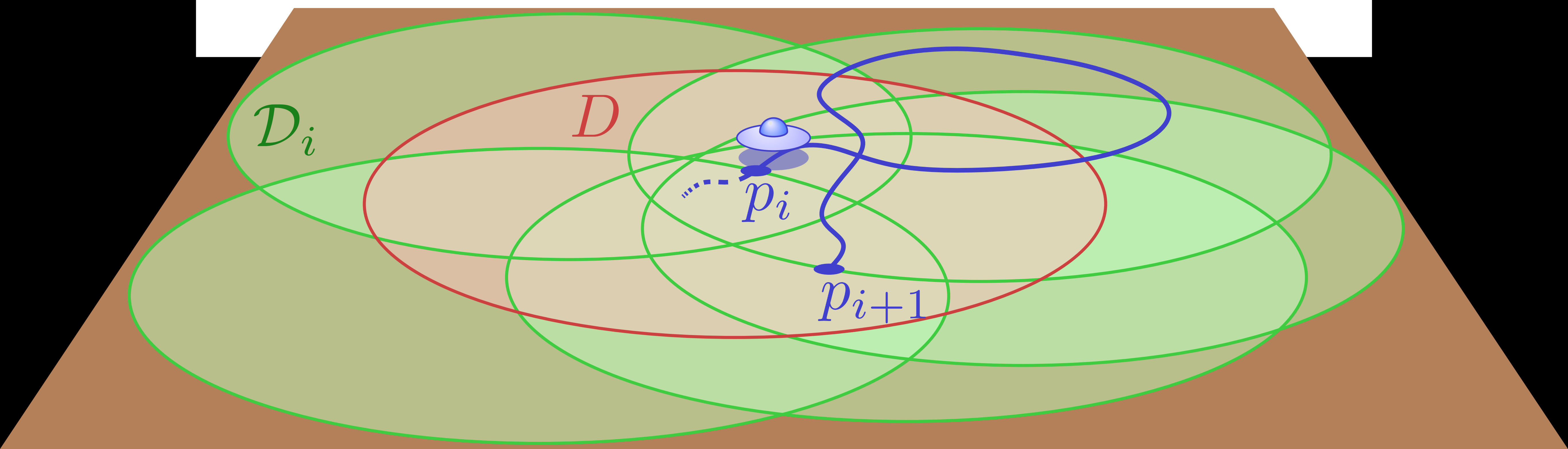
DETERMINISTIC ONLINE

- Algorithm
  - When at p_i, compute $\mathcal{D}_i$
  - Let D be "first" region of $\mathcal{D}_i$
  - When leaving a region, remove it from $\mathcal{D}_i$



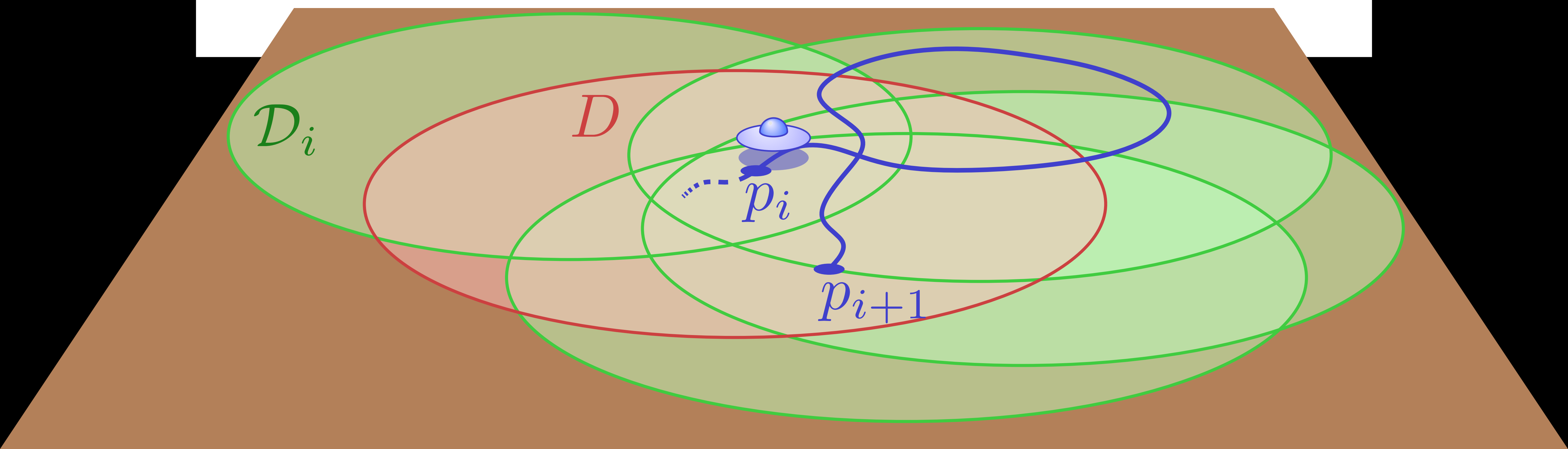
DETERMINISTIC ONLINE

- Algorithm
  - When at p_i, compute $\mathcal{D}_i$
  - Let D be “first” region of $\mathcal{D}_i$
  - When leaving a region, remove it from $\mathcal{D}_i$



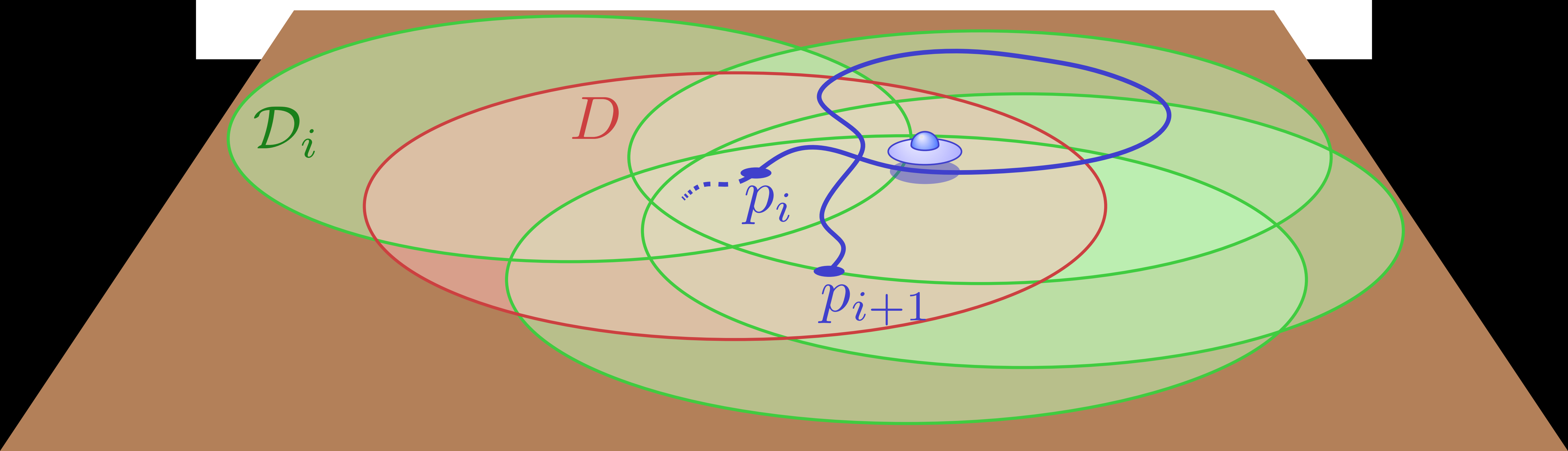
DETERMINISTIC ONLINE

- Algorithm
  - When at p_i, compute $\mathcal{D}_i$
  - Let D be "first" region of $\mathcal{D}_i$
  - When leaving a region, remove it from $\mathcal{D}_i$



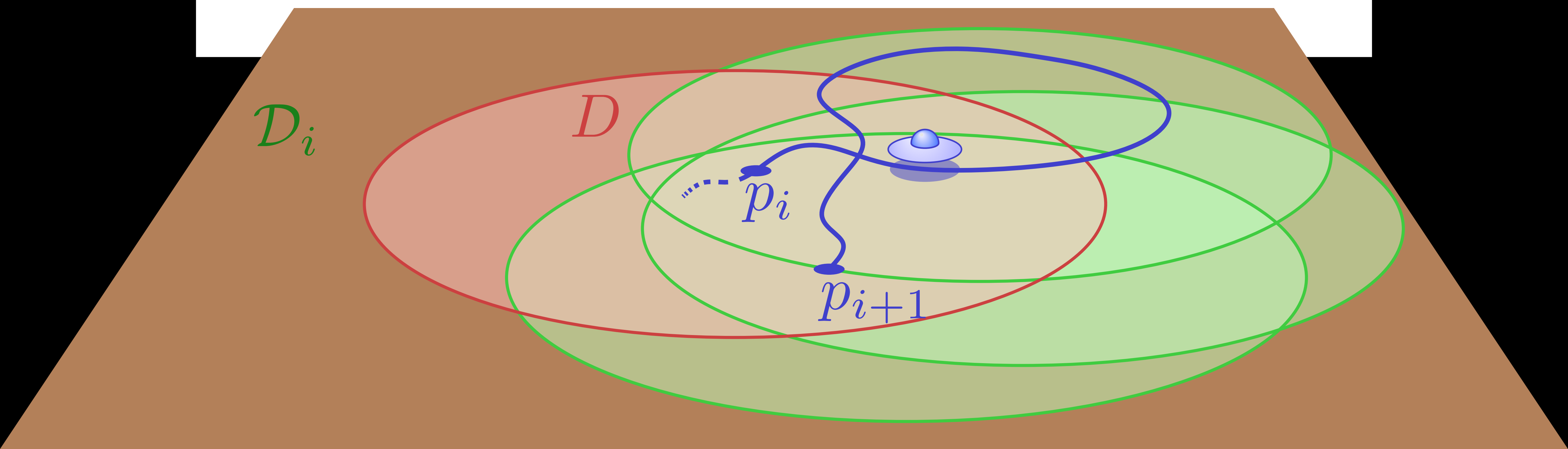
DETERMINISTIC ONLINE

- Algorithm
  - When at p_i, compute $\mathcal{D}_i$
  - Let D be "first" region of $\mathcal{D}_i$
  - When leaving a region, remove it from $\mathcal{D}_i$



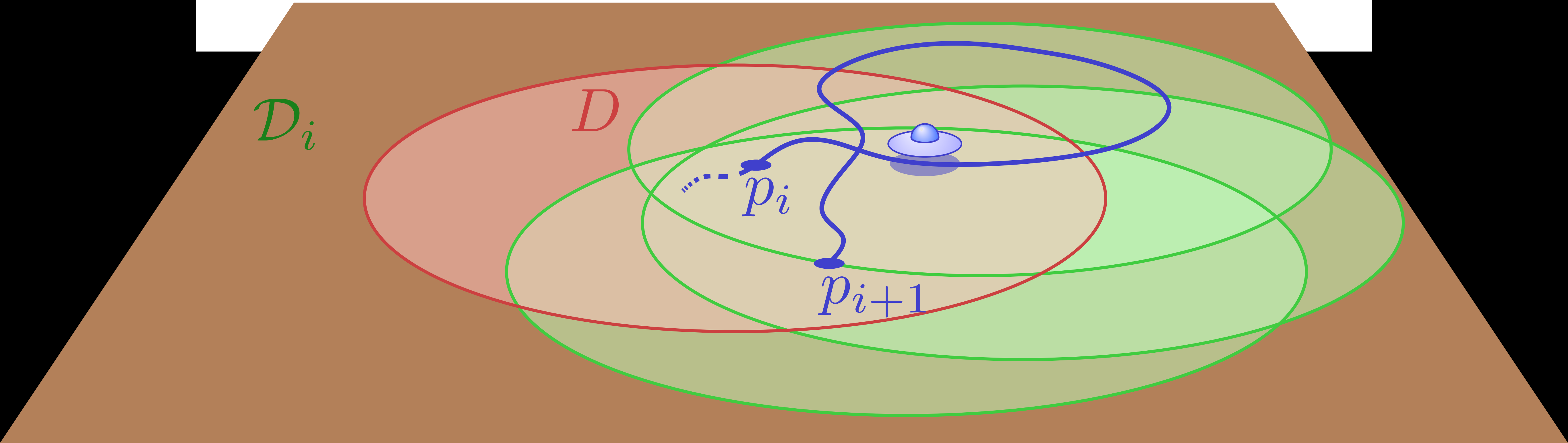
DETERMINISTIC ONLINE

- Algorithm
  - When at p_i, compute $\mathcal{D}_i$
  - Let D be "first" region of $\mathcal{D}_i$
  - When leaving a region, remove it from $\mathcal{D}_i$



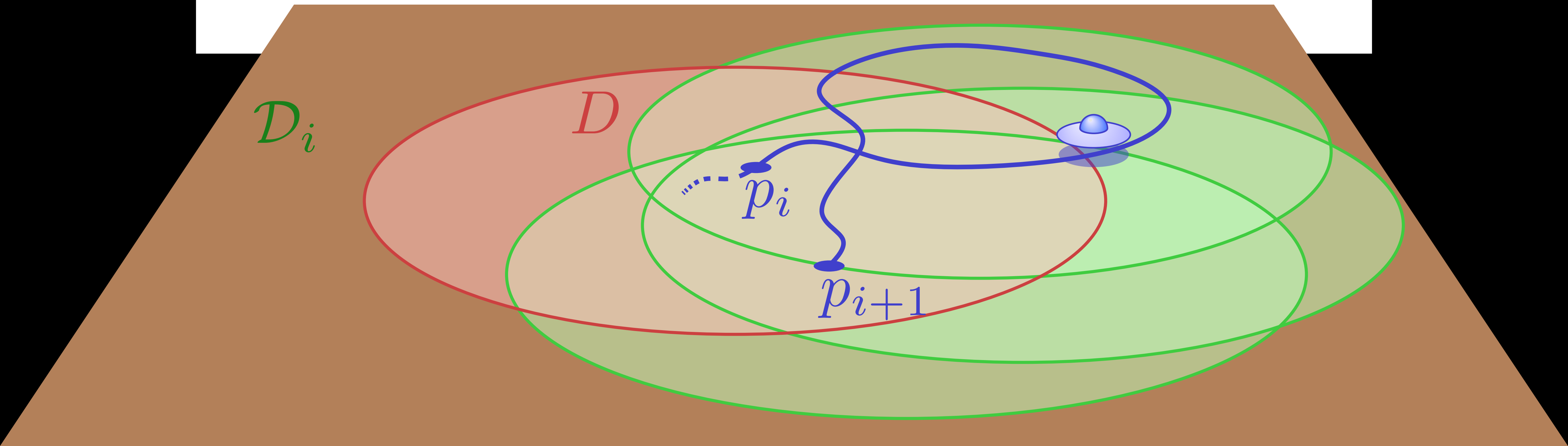
DETERMINISTIC ONLINE

- Algorithm
  - When at p_i, compute $\mathcal{D}_i$
  - Let D be “first” region of $\mathcal{D}_i$
  - When leaving a region, remove it from $\mathcal{D}_i$
  - When leaving D and $\mathcal{D}_i$ is not empty yet, pick “next” region *from* $\mathcal{D}_i$



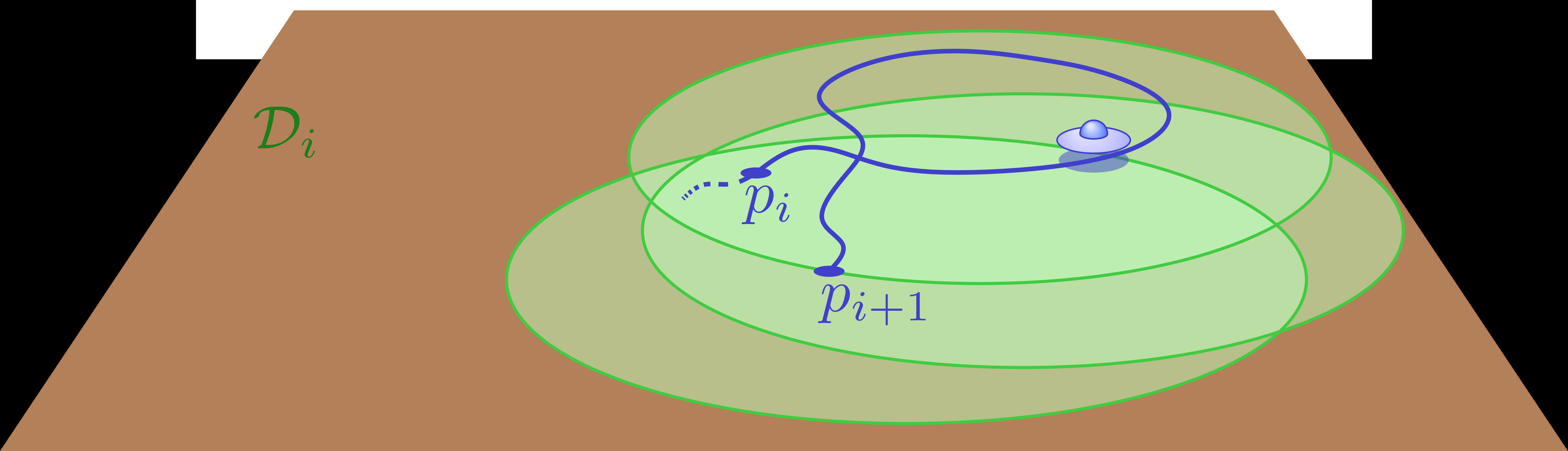
DETERMINISTIC ONLINE

- Algorithm
  - When at p_i, compute $\mathcal{D}_i$
  - Let D be "first" region of $\mathcal{D}_i$
  - When leaving a region, remove it from $\mathcal{D}_i$
  - When leaving D and $\mathcal{D}_i$ is not empty yet, pick "next" region *from* $\mathcal{D}_i$



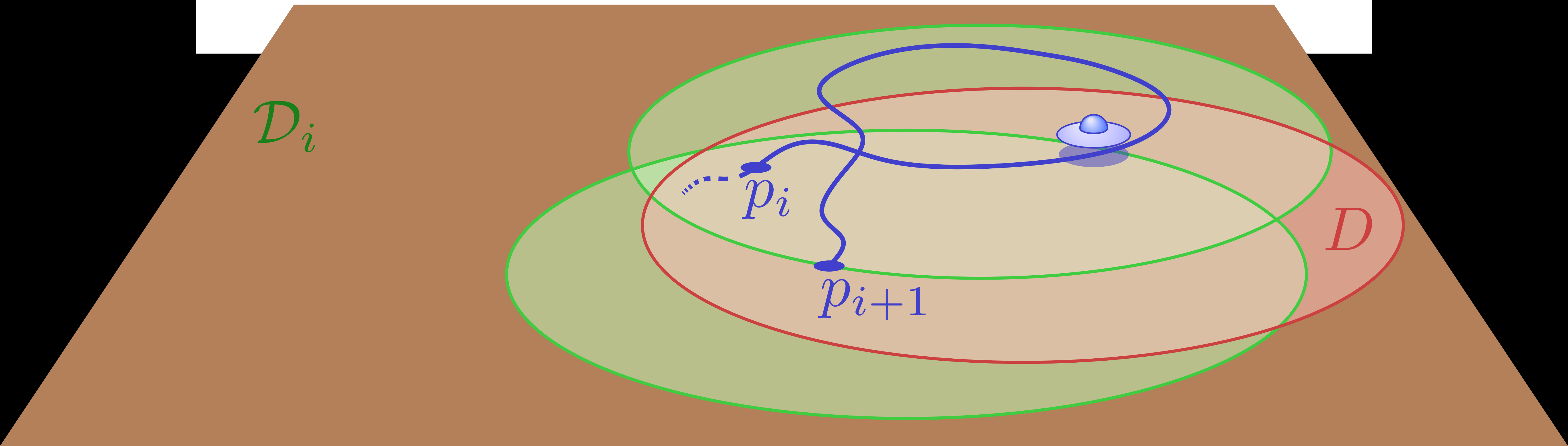
DETERMINISTIC ONLINE

- Algorithm
  - When at p_i, compute $\mathcal{D}_i$
  - Let D be “first” region of $\mathcal{D}_i$
  - When leaving a region, remove it from $\mathcal{D}_i$
  - When leaving D and $\mathcal{D}_i$ is not empty yet, pick “next” region *from* $\mathcal{D}_i$



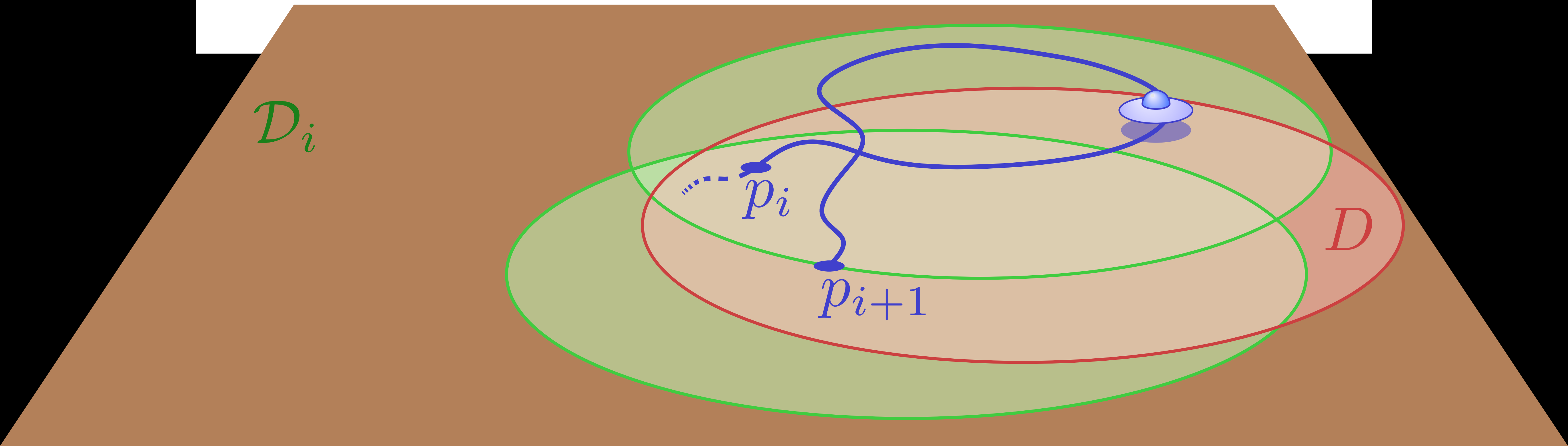
DETERMINISTIC ONLINE

- Algorithm
  - When at p_i, compute $\mathcal{D}_i$
  - Let D be "first" region of $\mathcal{D}_i$
  - When leaving a region, remove it from $\mathcal{D}_i$
  - When leaving D and $\mathcal{D}_i$ is not empty yet, pick "next" region *from* $\mathcal{D}_i$



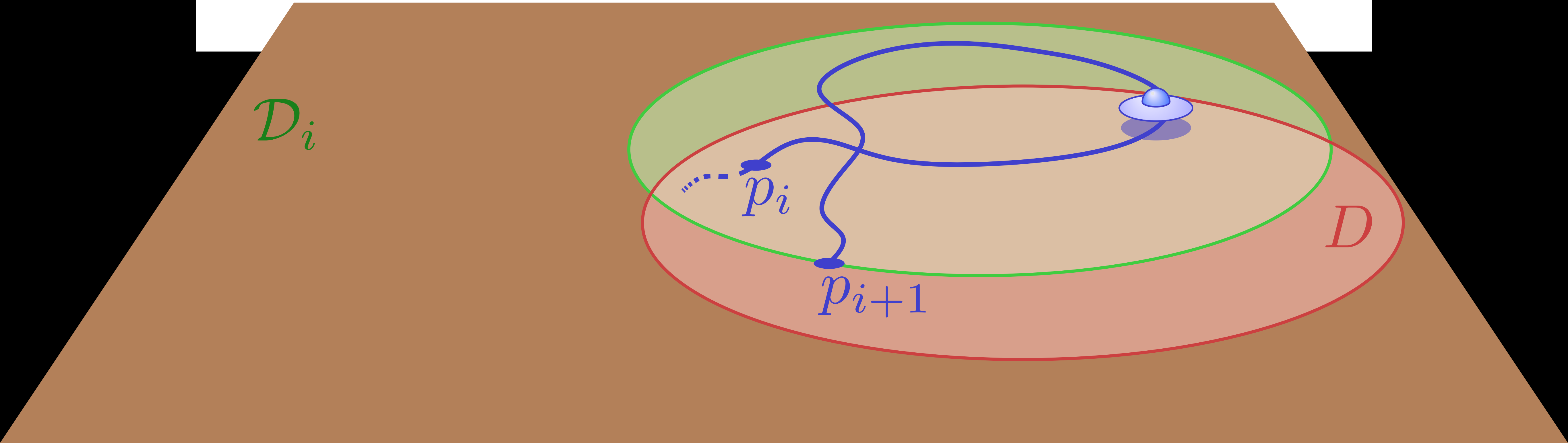
DETERMINISTIC ONLINE

- Algorithm
  - When at p_i, compute $\mathcal{D}_i$
  - Let D be "first" region of $\mathcal{D}_i$
  - When leaving a region, remove it from $\mathcal{D}_i$
  - When leaving D and $\mathcal{D}_i$ is not empty yet, pick "next" region *from* $\mathcal{D}_i$



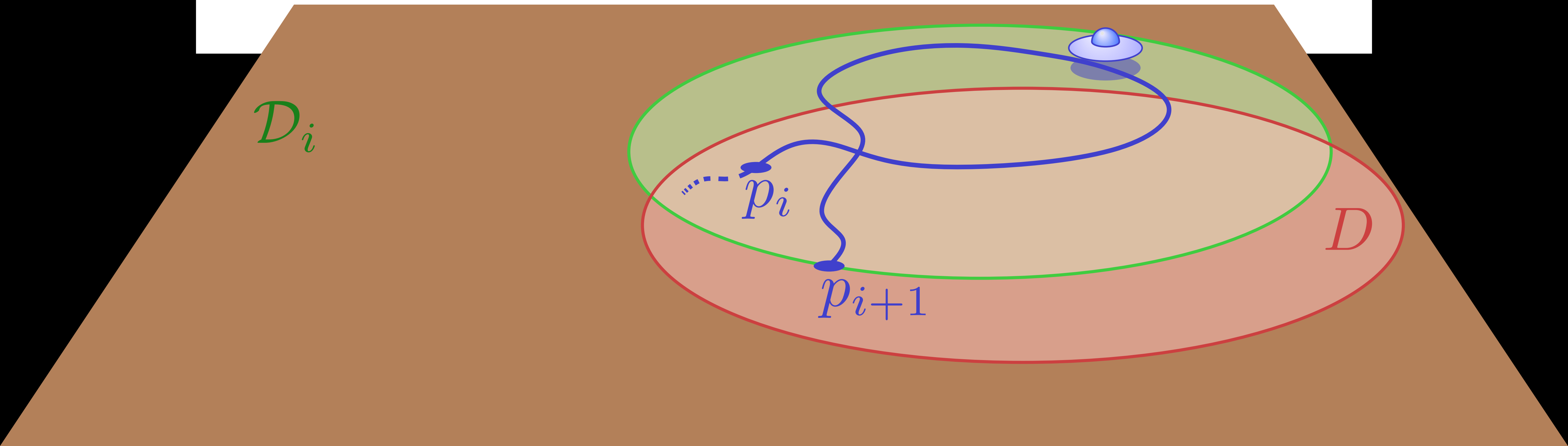
DETERMINISTIC ONLINE

- Algorithm
  - When at p_i, compute $\mathcal{D}_i$
  - Let D be "first" region of $\mathcal{D}_i$
  - When leaving a region, remove it from $\mathcal{D}_i$
  - When leaving D and $\mathcal{D}_i$ is not empty yet, pick "next" region *from* $\mathcal{D}_i$



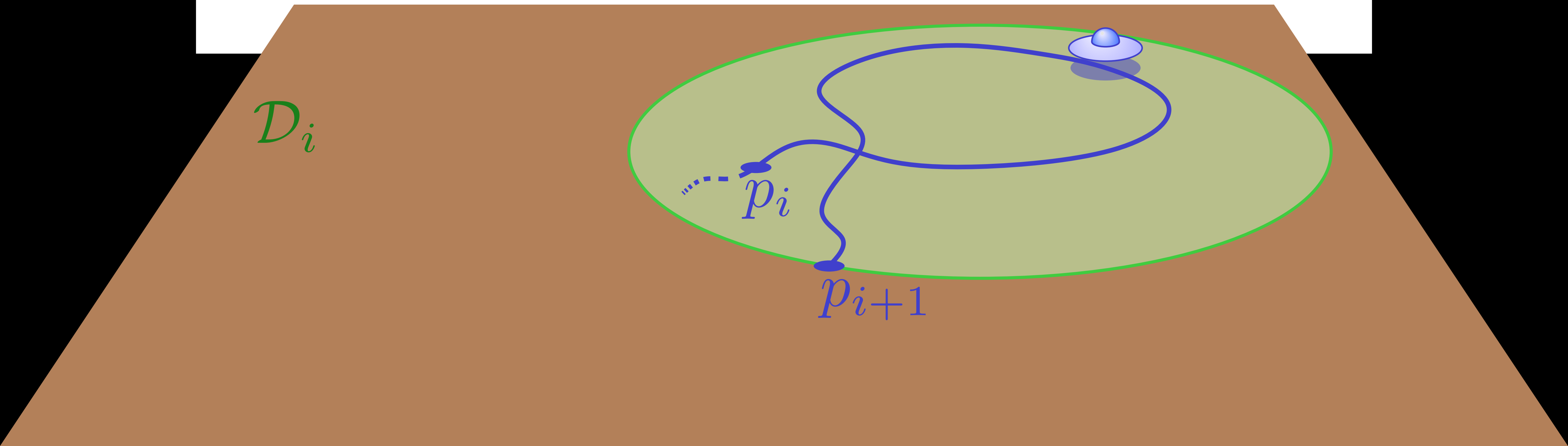
DETERMINISTIC ONLINE

- Algorithm
  - When at p_i, compute $\mathcal{D}_i$
  - Let D be "first" region of $\mathcal{D}_i$
  - When leaving a region, remove it from $\mathcal{D}_i$
  - When leaving D and $\mathcal{D}_i$ is not empty yet, pick "next" region *from* $\mathcal{D}_i$



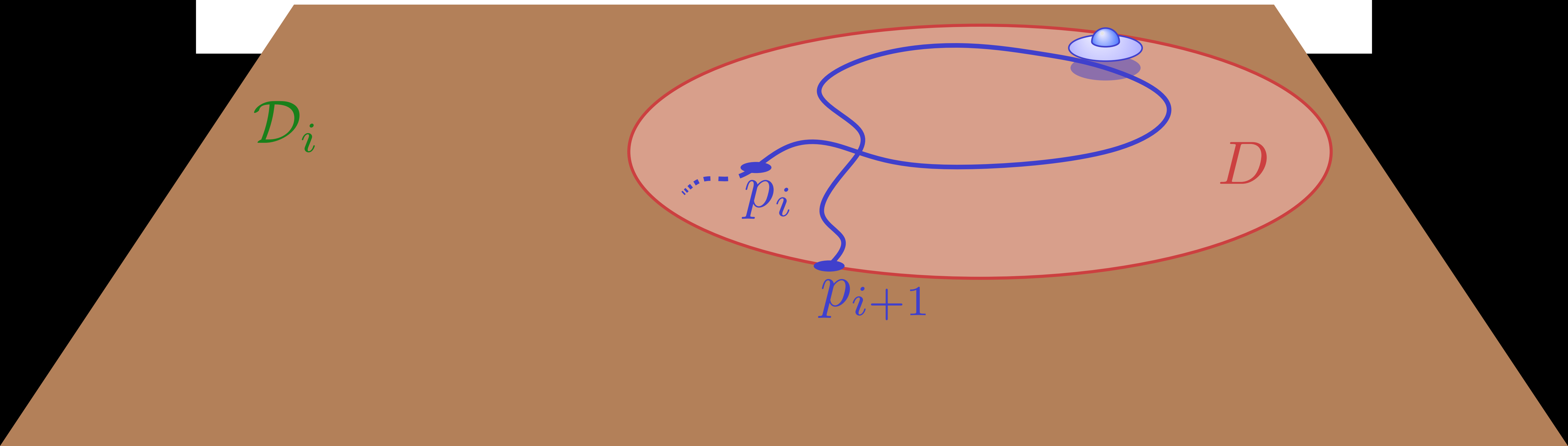
DETERMINISTIC ONLINE

- Algorithm
  - When at p_i, compute $\mathcal{D}_i$
  - Let D be "first" region of $\mathcal{D}_i$
  - When leaving a region, remove it from $\mathcal{D}_i$
  - When leaving D and $\mathcal{D}_i$ is not empty yet, pick "next" region *from* $\mathcal{D}_i$



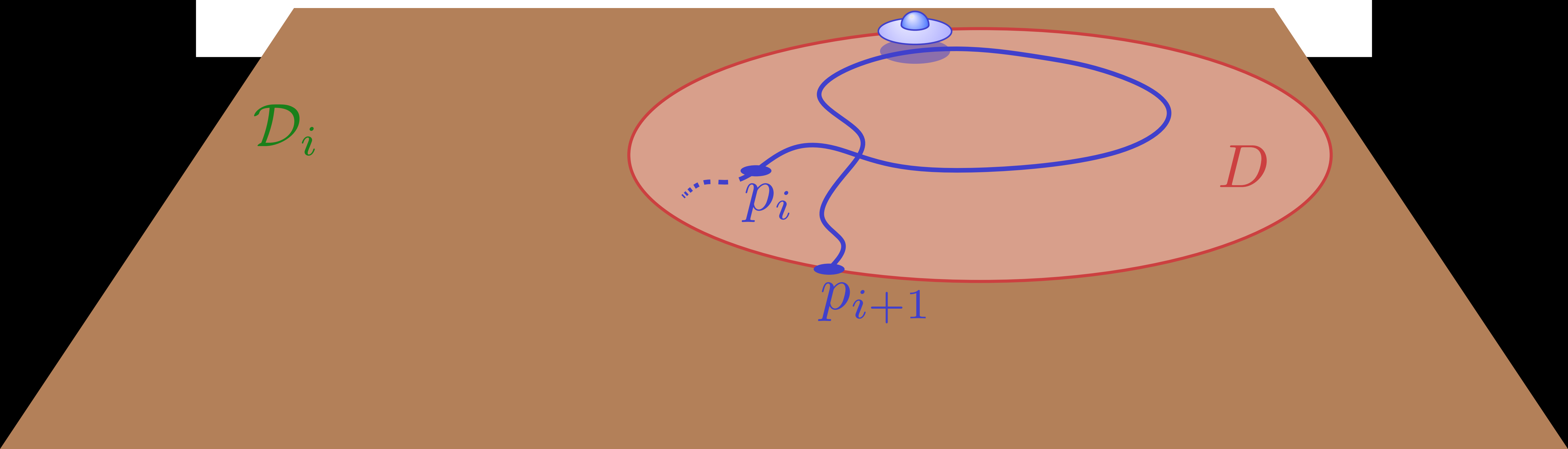
DETERMINISTIC ONLINE

- Algorithm
  - When at p_i, compute $\mathcal{D}_i$
  - Let D be “first” region of $\mathcal{D}_i$
  - When leaving a region, remove it from $\mathcal{D}_i$
  - When leaving D and $\mathcal{D}_i$ is not empty yet, pick “next” region *from* $\mathcal{D}_i$



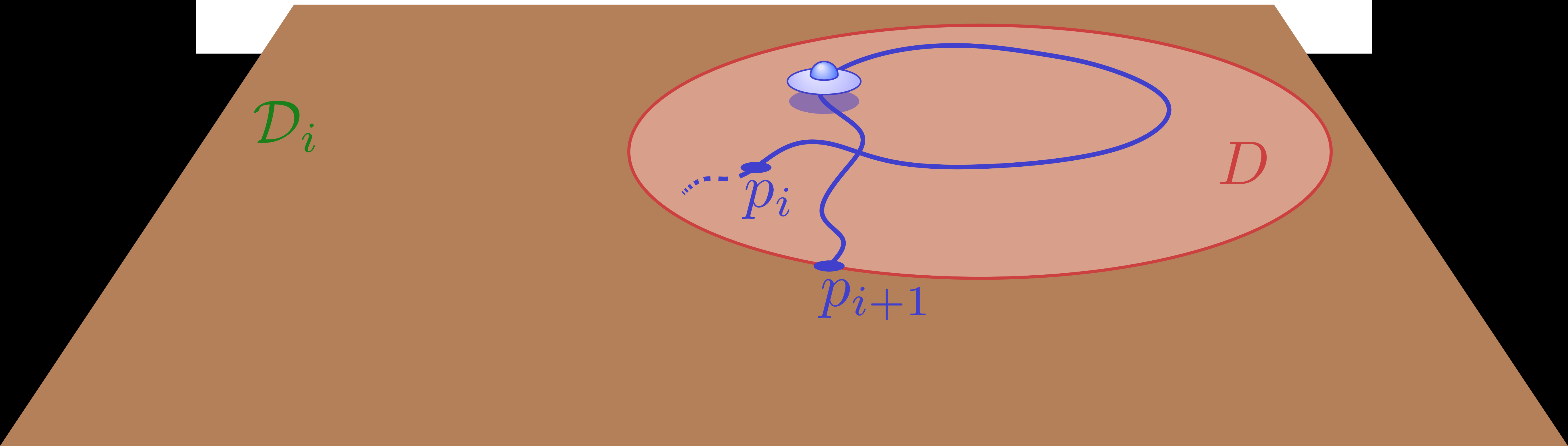
DETERMINISTIC ONLINE

- Algorithm
  - When at p_i, compute $\mathcal{D}_i$
  - Let D be "first" region of $\mathcal{D}_i$
  - When leaving a region, remove it from $\mathcal{D}_i$
  - When leaving D and $\mathcal{D}_i$ is not empty yet, pick "next" region *from* $\mathcal{D}_i$



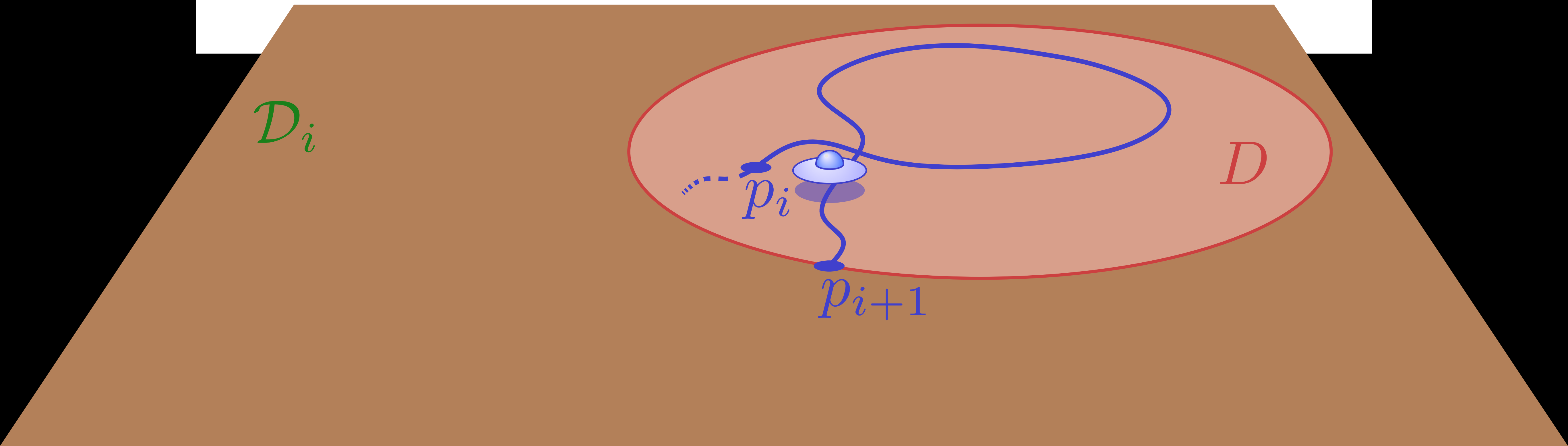
DETERMINISTIC ONLINE

- Algorithm
  - When at p_i, compute $\mathcal{D}_i$
  - Let D be "first" region of $\mathcal{D}_i$
  - When leaving a region, remove it from $\mathcal{D}_i$
  - When leaving D and $\mathcal{D}_i$ is not empty yet, pick "next" region *from* $\mathcal{D}_i$



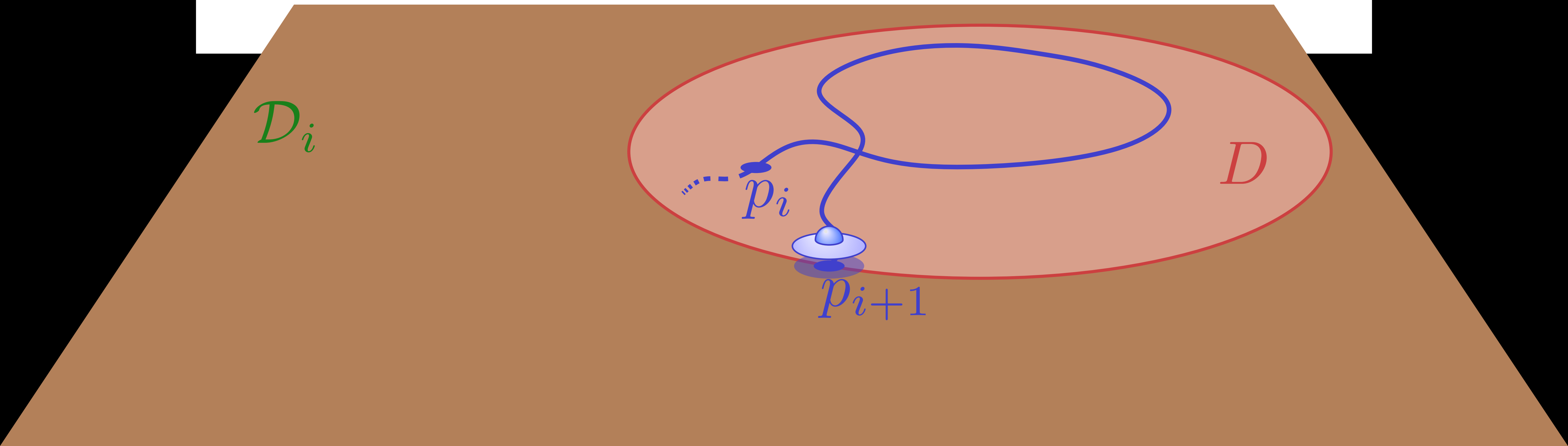
DETERMINISTIC ONLINE

- Algorithm
  - When at p_i, compute $\mathcal{D}_i$
  - Let D be "first" region of $\mathcal{D}_i$
  - When leaving a region, remove it from $\mathcal{D}_i$
  - When leaving D and $\mathcal{D}_i$ is not empty yet, pick "next" region *from* $\mathcal{D}_i$



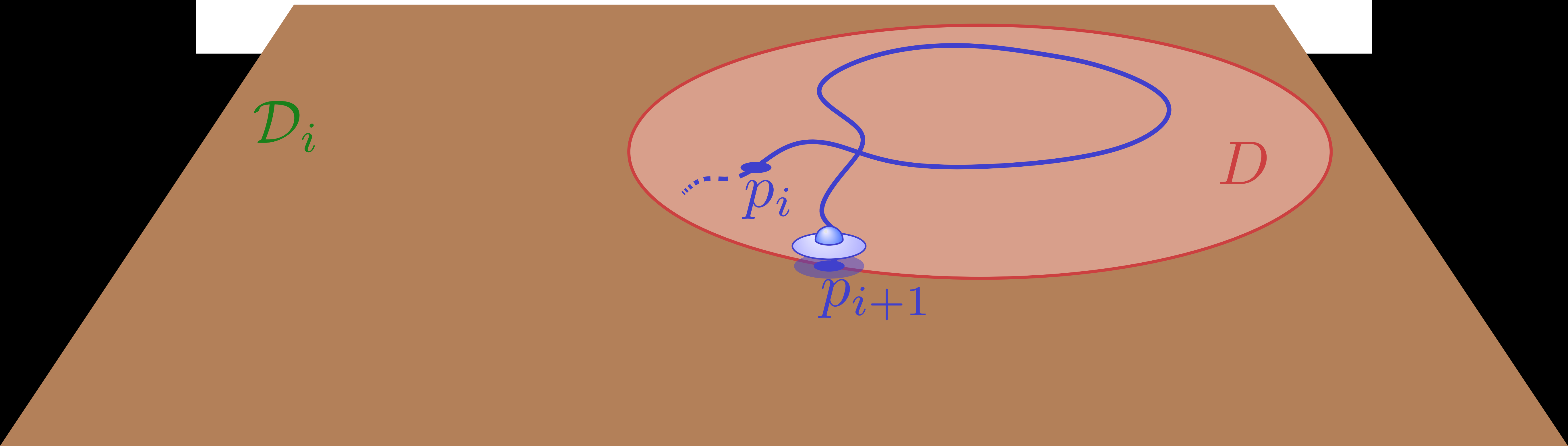
DETERMINISTIC ONLINE

- Algorithm
  - When at p_i, compute $\mathcal{D}_i$
  - Let D be "first" region of $\mathcal{D}_i$
  - When leaving a region, remove it from $\mathcal{D}_i$
  - When leaving D and $\mathcal{D}_i$ is not empty yet, pick "next" region *from* $\mathcal{D}_i$



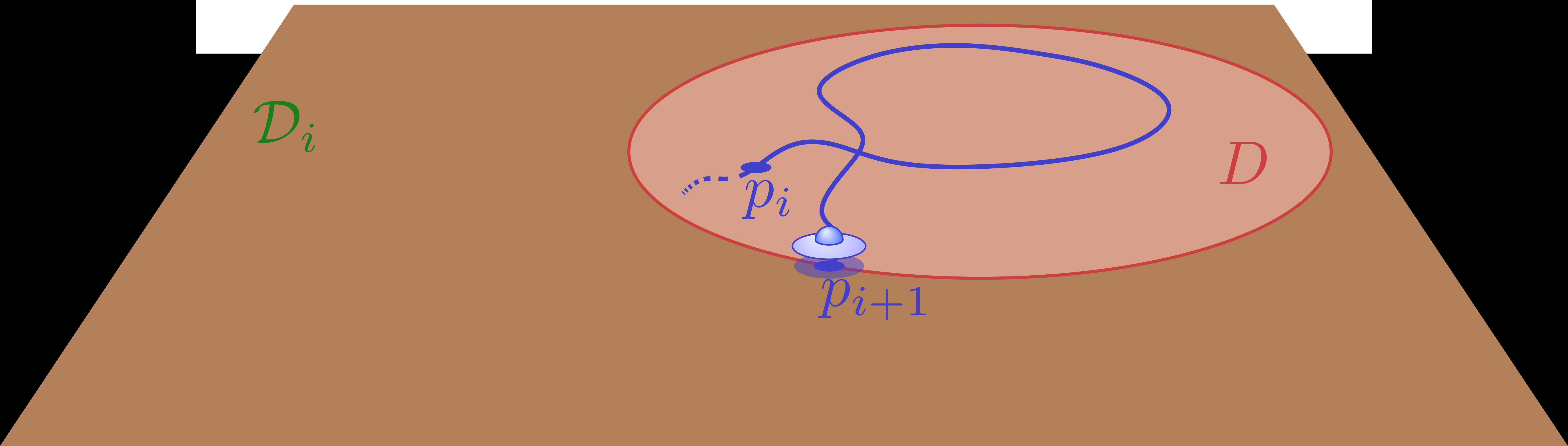
DETERMINISTIC ONLINE

- Algorithm
  - When at p_i, compute $\mathcal{D}_i$
  - Let D be "first" region of $\mathcal{D}_i$
  - When leaving a region, remove it from $\mathcal{D}_i$
  - When leaving D and $\mathcal{D}_i$ is not empty yet, pick "next" region *from* $\mathcal{D}_i$
  - When $\mathcal{D}_i$ is empty, we found p_{i+1}



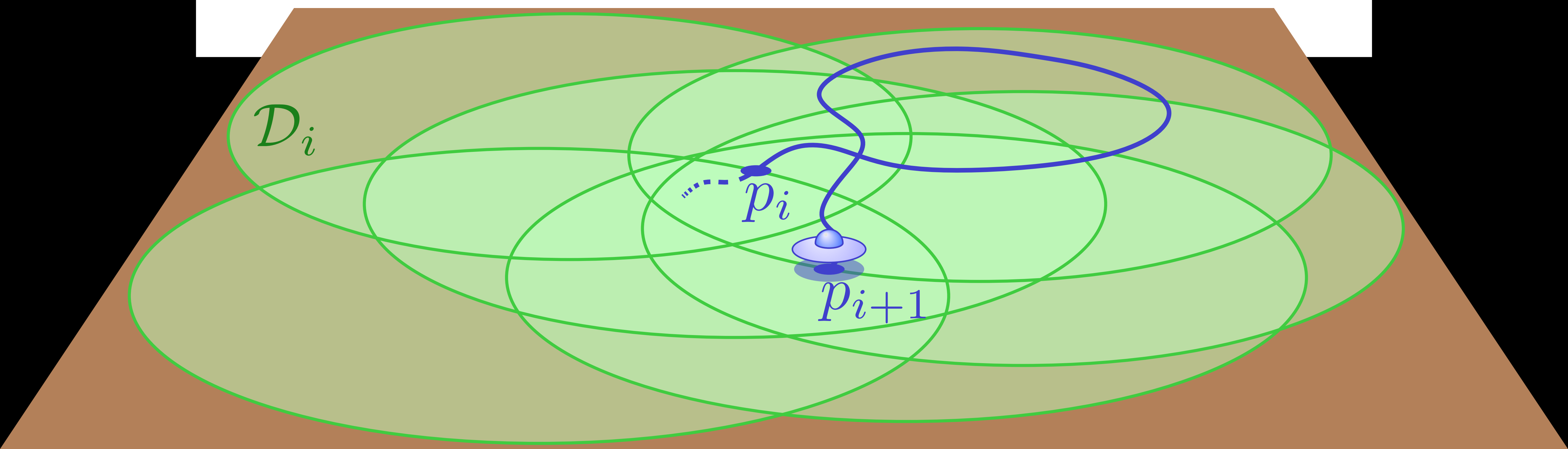
DETERMINISTIC ONLINE

- Algorithm
  - When at p_i, compute $\mathcal{D}_i$
  - Let D be "first" region of $\mathcal{D}_i$
  - When leaving a region, remove it from $\mathcal{D}_i$
  - When leaving D and $\mathcal{D}_i$ is not empty yet, pick "next" region *from* $\mathcal{D}_i$
  - When $\mathcal{D}_i$ is empty, we found p_{i+1}
- Analysis



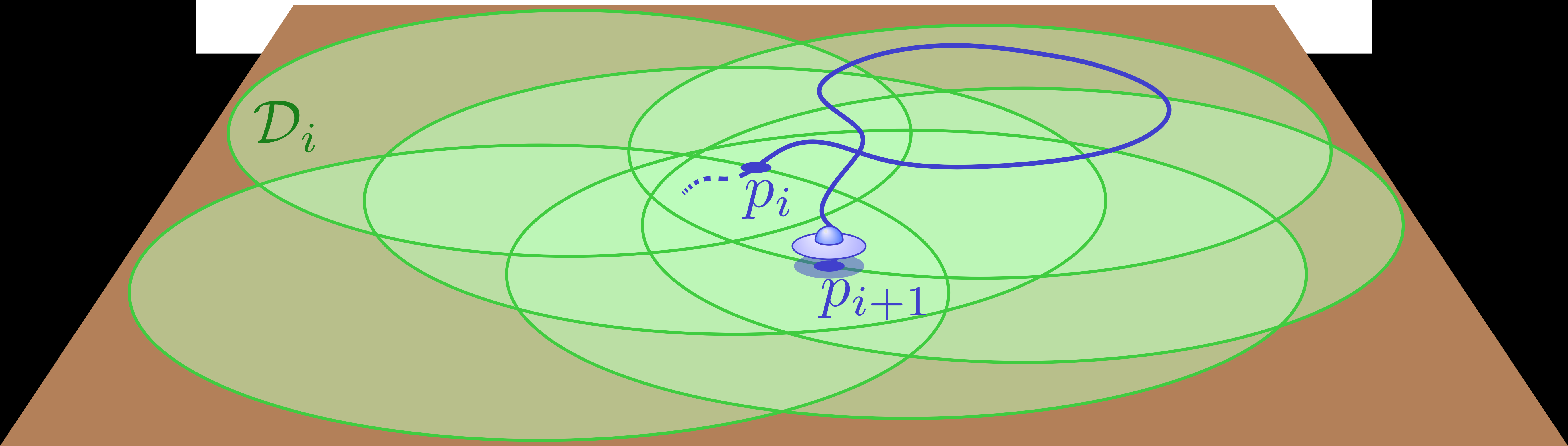
DETERMINISTIC ONLINE

- Algorithm
  - When at p_i, compute $\mathcal{D}_i$
  - Let D be "first" region of $\mathcal{D}_i$
  - When leaving a region, remove it from $\mathcal{D}_i$
  - When leaving D and $\mathcal{D}_i$ is not empty yet, pick "next" region *from* $\mathcal{D}_i$
  - When $\mathcal{D}_i$ is empty, we found p_{i+1}
- Analysis



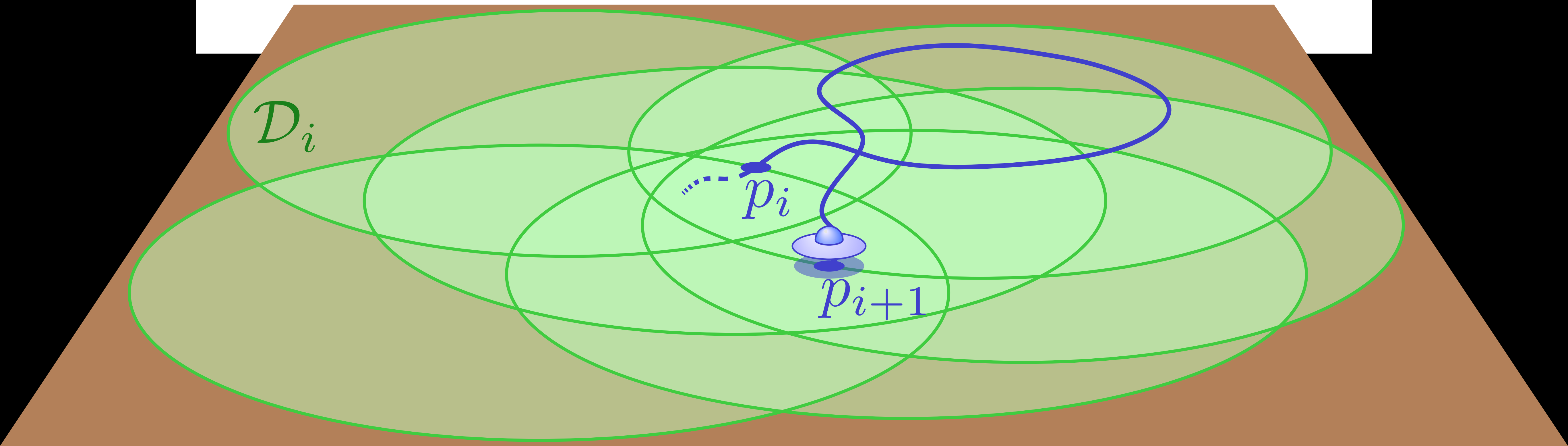
DETERMINISTIC ONLINE

- Algorithm
  - When at p_i, compute $\mathcal{D}_i$
  - Let D be "first" region of $\mathcal{D}_i$
  - When leaving a region, remove it from $\mathcal{D}_i$
  - When leaving D and $\mathcal{D}_i$ is not empty yet, pick "next" region *from* $\mathcal{D}_i$
  - When $\mathcal{D}_i$ is empty, we found p_{i+1}
- Analysis
  - At most $|\mathcal{D}_i| \leq \Delta$ handovers

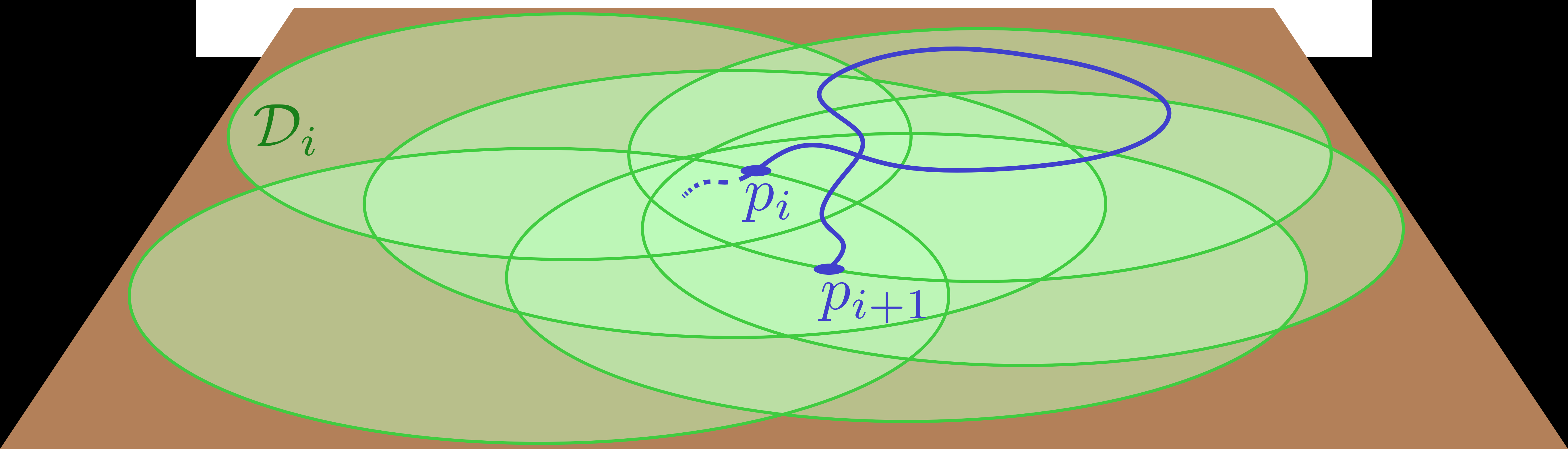


DETERMINISTIC ONLINE

- Algorithm
  - When at p_i, compute $\mathcal{D}_i$
  - Let D be "first" region of $\mathcal{D}_i$
  - When leaving a region, remove it from $\mathcal{D}_i$
  - When leaving D and $\mathcal{D}_i$ is not empty yet, pick "next" region *from* $\mathcal{D}_i$
  - When $\mathcal{D}_i$ is empty, we found p_{i+1}
- Analysis
  - At most $|\mathcal{D}_i| \leq \Delta$ handovers
  - Competitive ratio is Δ

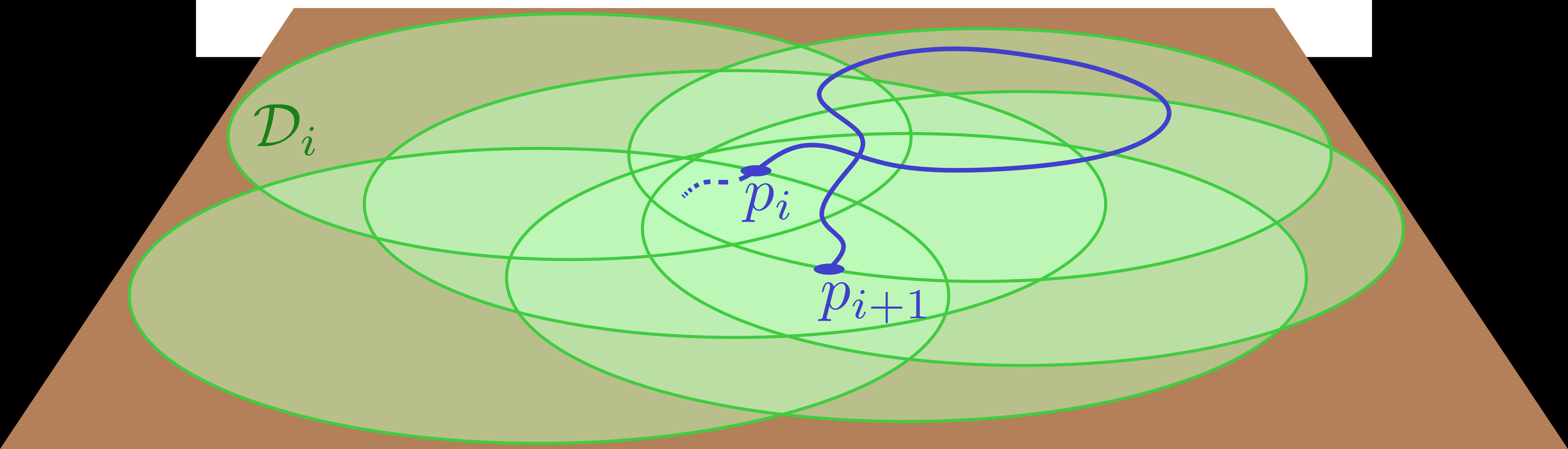


RANDOMISED ONLINE



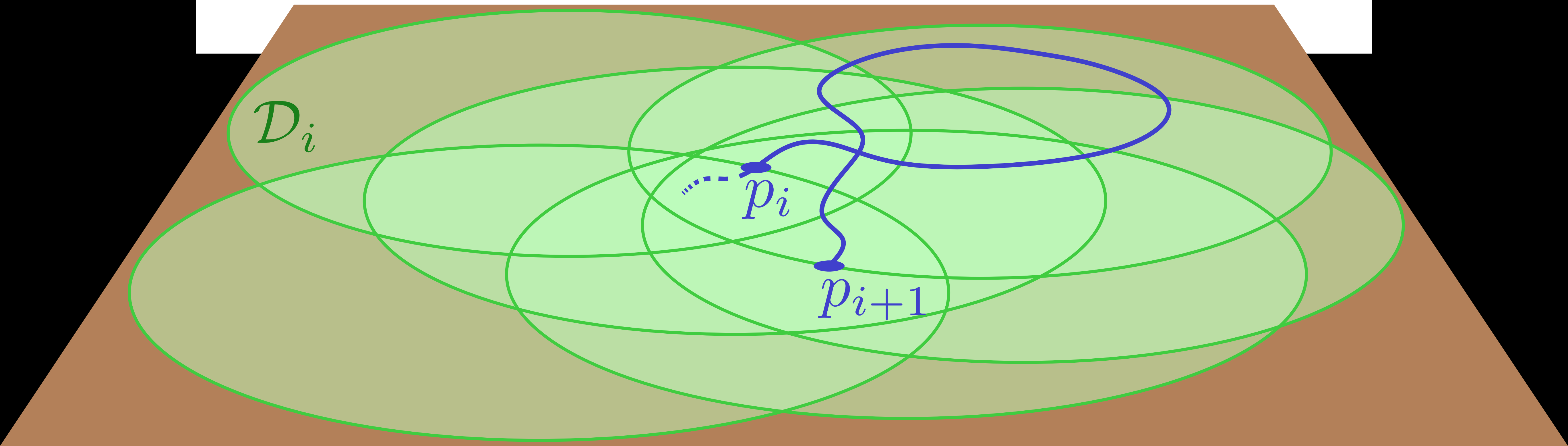
RANDOMISED ONLINE

- Algorithm



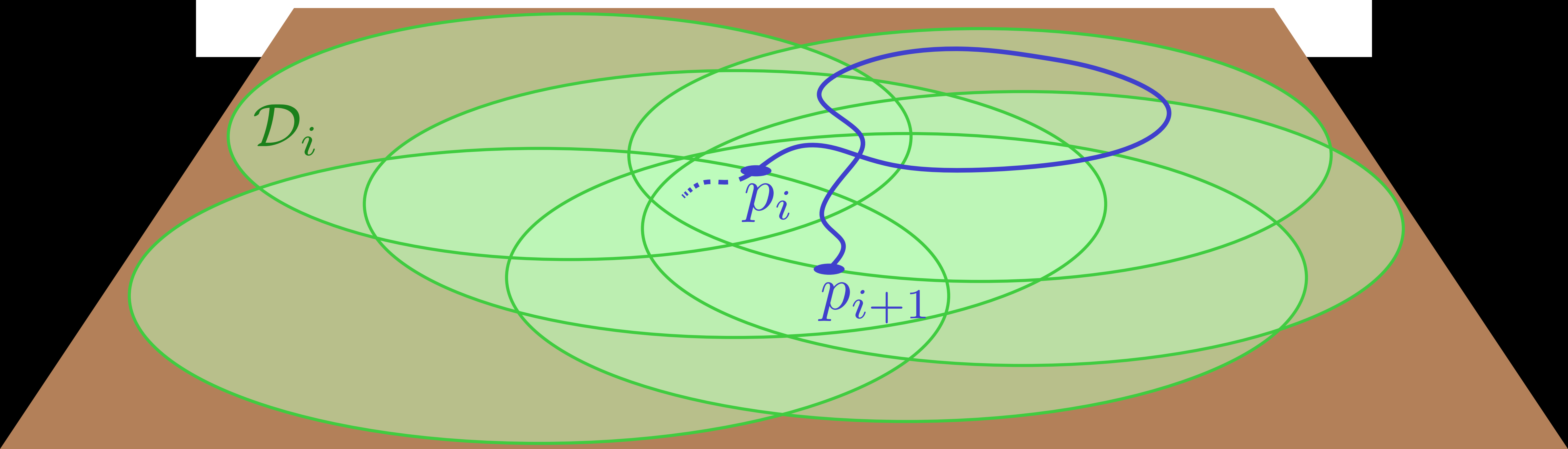
RANDOMISED ONLINE

- Algorithm
  - Almost the same as deterministic



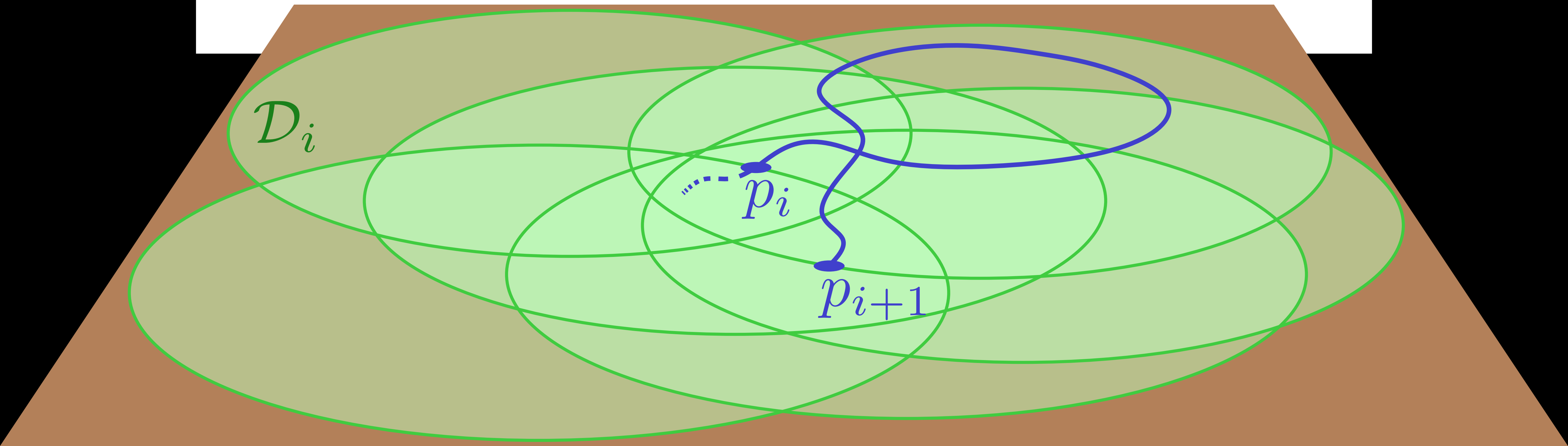
RANDOMISED ONLINE

- Algorithm
  - Almost the same as deterministic
  - Just choose $D \in \mathcal{D}_i$ randomly



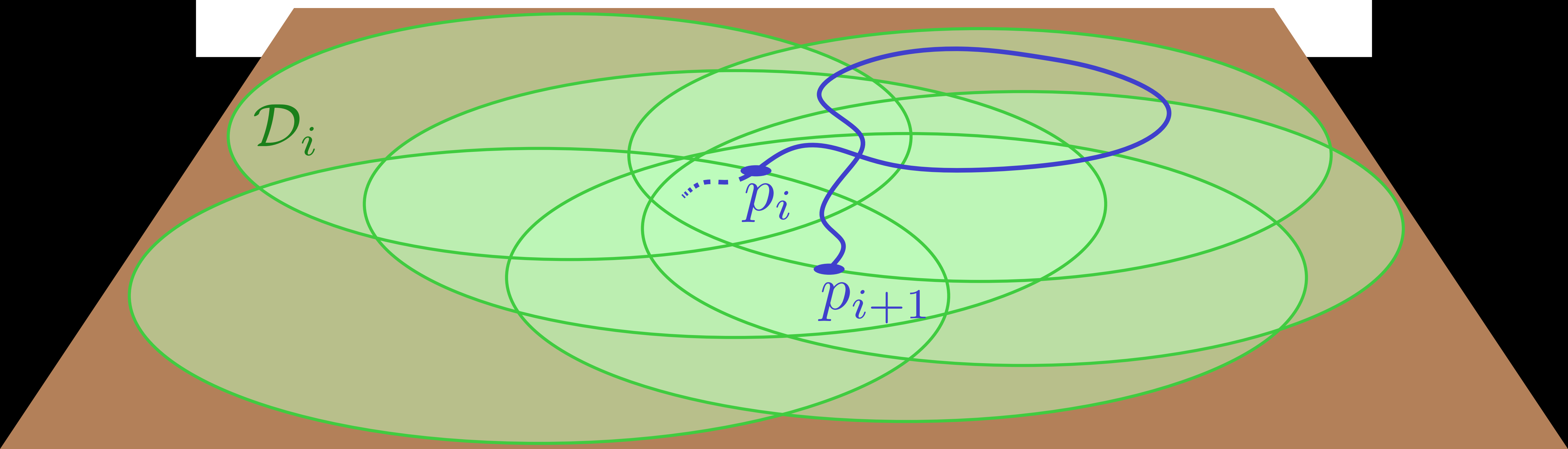
RANDOMISED ONLINE

- Algorithm
  - Almost the same as deterministic
  - Just choose $D \in \mathcal{D}_i$ randomly
- Analysis



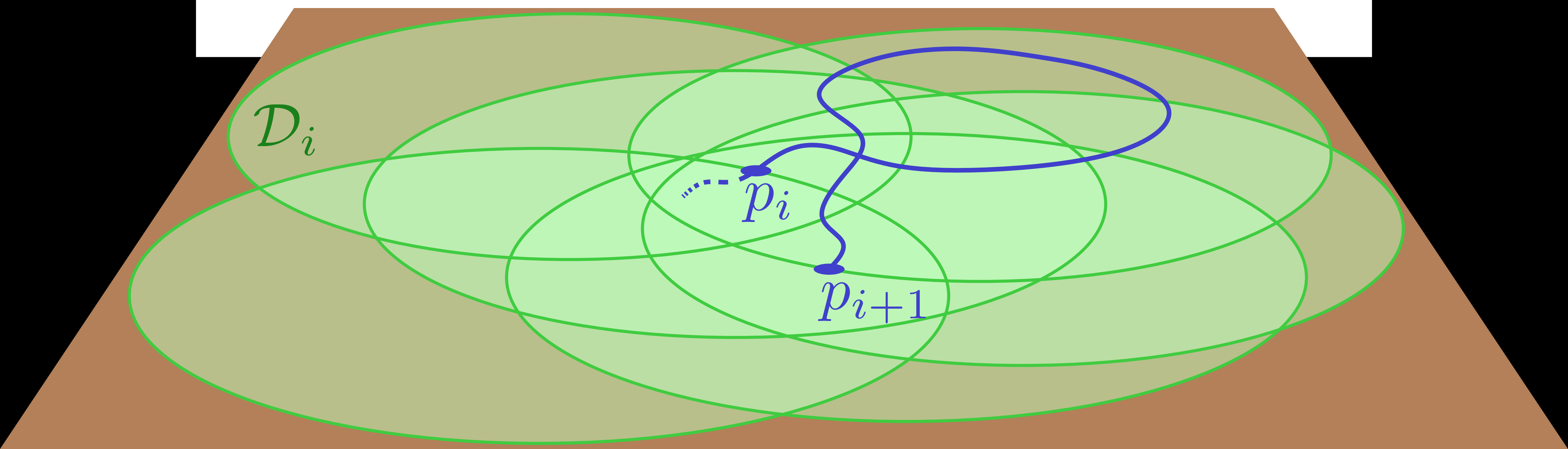
RANDOMISED ONLINE

- Algorithm
  - Almost the same as deterministic
  - Just choose $D \in \mathcal{D}_i$ randomly
- Analysis
  - Sort $\mathcal{D}_i = D_1, \ldots, D_k$ by time object leaves them



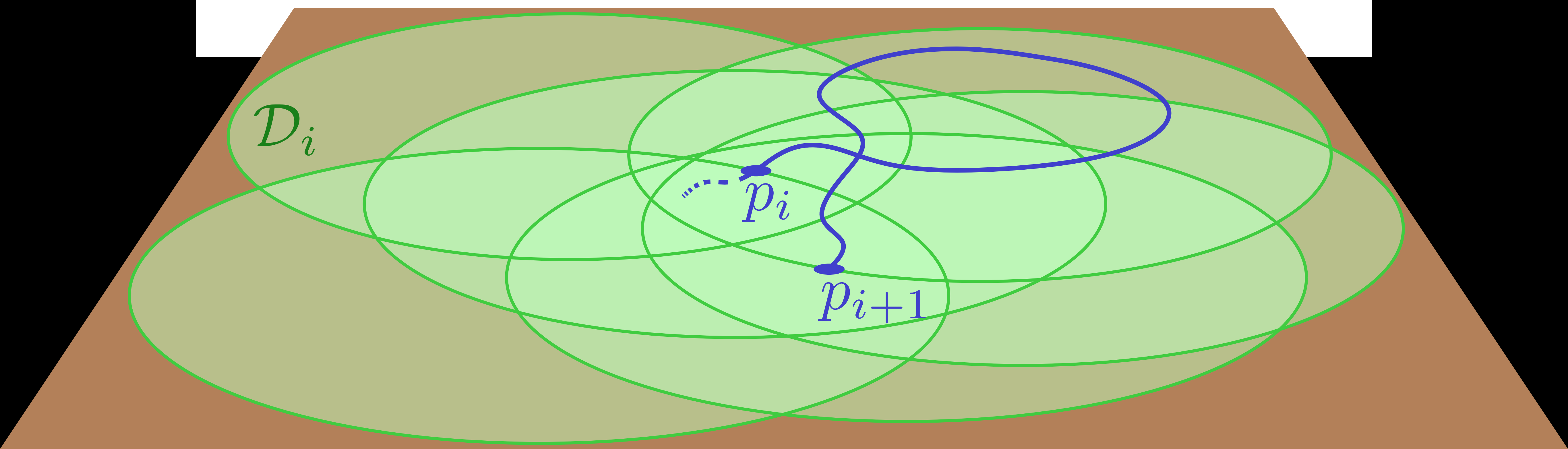
RANDOMISED ONLINE

- Algorithm
  - Almost the same as deterministic
  - Just choose $D \in \mathcal{D}_i$ randomly
- Analysis
  - Sort $\mathcal{D}_i = D_1, \ldots, D_k$ by time object leaves them
  - Take random increasing sequence of indices



RANDOMISED ONLINE

- Algorithm
  - Almost the same as deterministic
  - Just choose $D \in \mathcal{D}_i$ randomly
- Analysis
  - Sort $\mathcal{D}_i = D_1, \ldots, D_k$ by time object leaves them
  - Take random increasing sequence of indices
  - Expected lenght $\log k \leq \log \Delta$



RANDOMISED ONLINE

RANDOMISED ONLINE

- Lower bound construction

RANDOMISED ONLINE

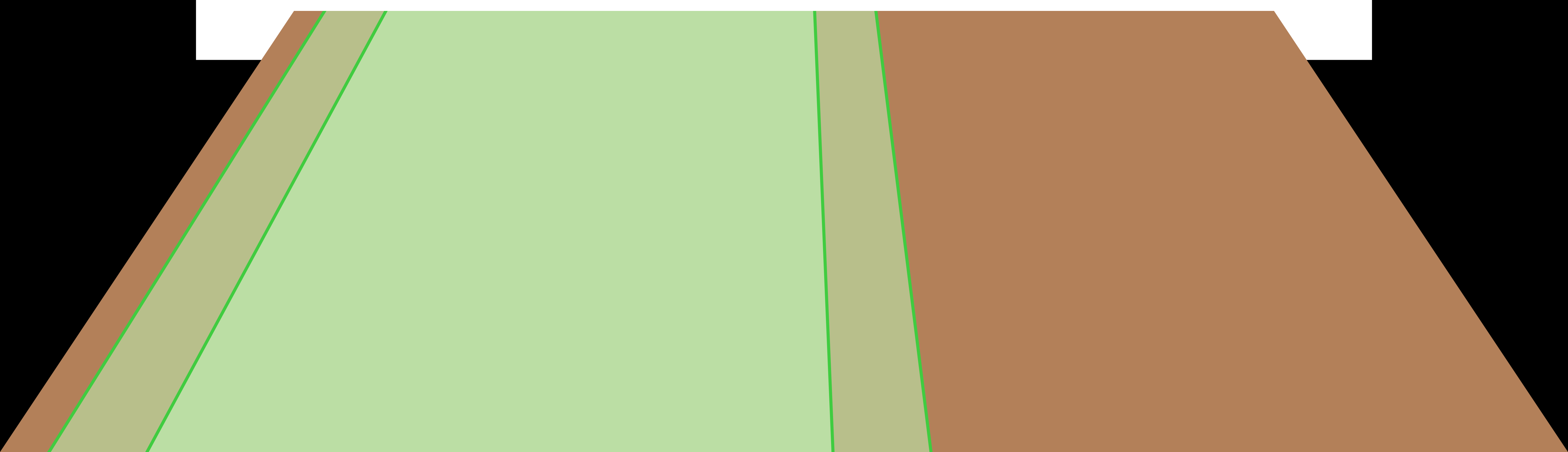
- Lower bound construction
  - Δ partially overlapping intervals

RANDOMISED ONLINE

- Lower bound construction
  - Δ partially overlapping intervals

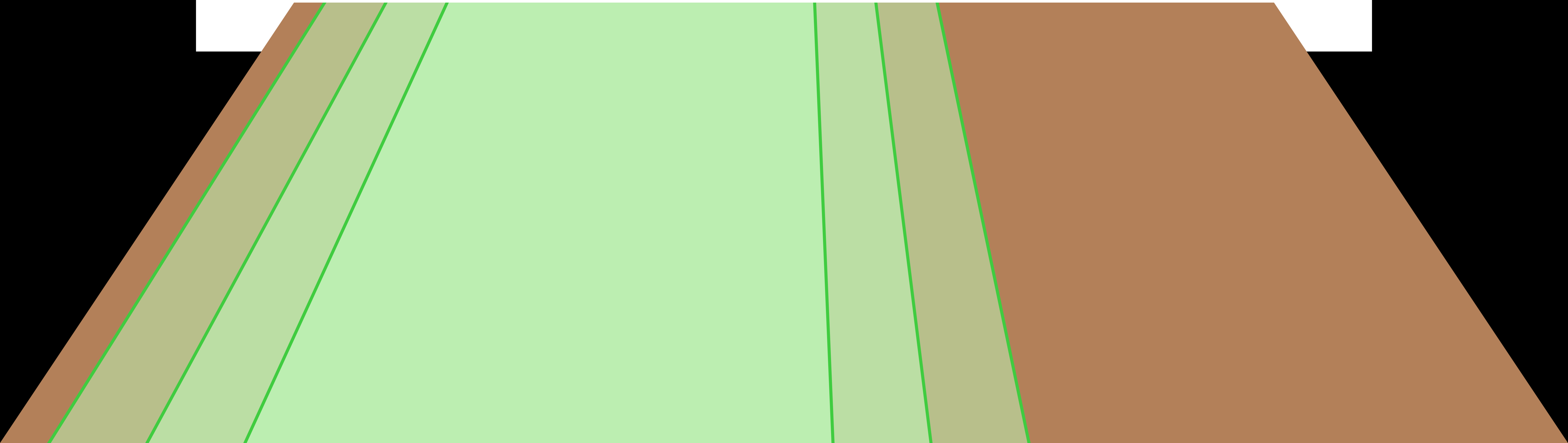
RANDOMISED ONLINE

- Lower bound construction
  - Δ partially overlapping intervals



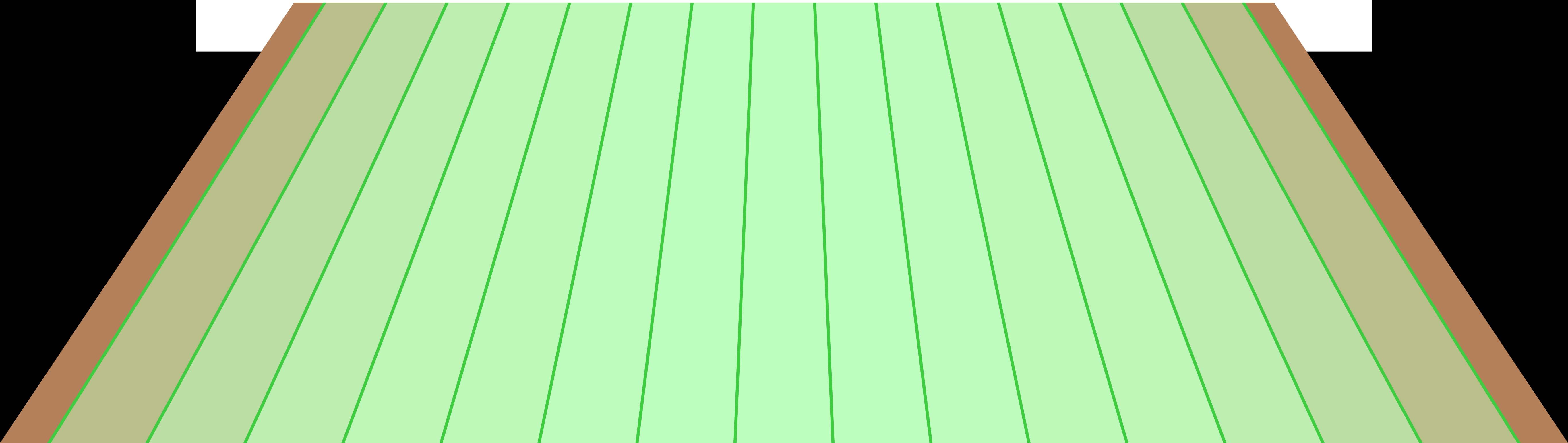
RANDOMISED ONLINE

- Lower bound construction
  - Δ partially overlapping intervals



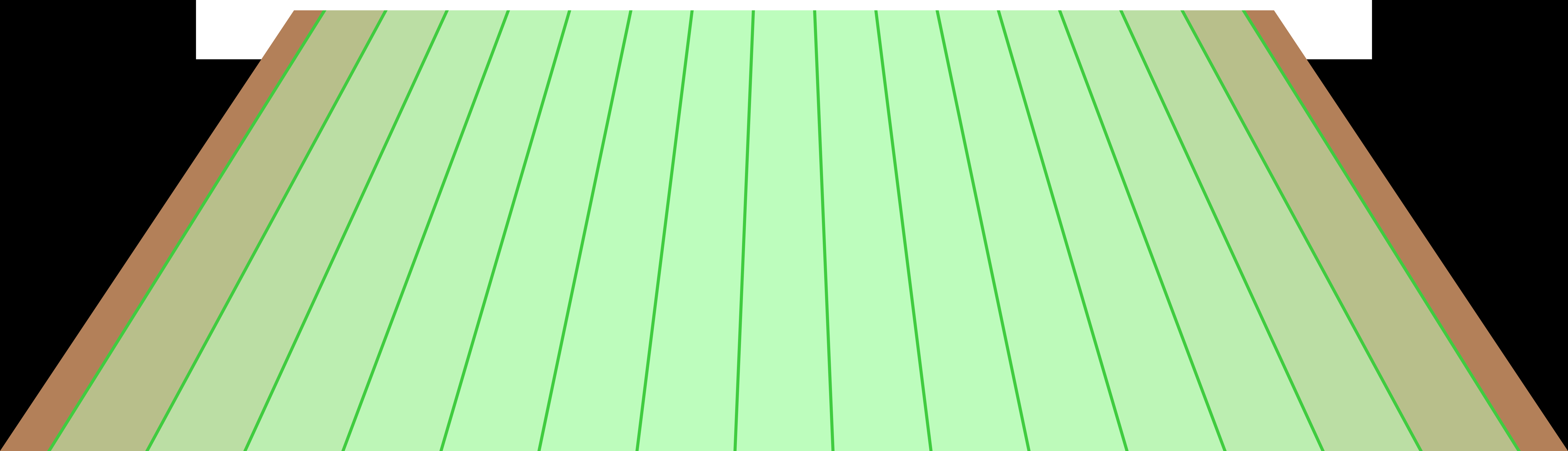
RANDOMISED ONLINE

- Lower bound construction
  - Δ partially overlapping intervals



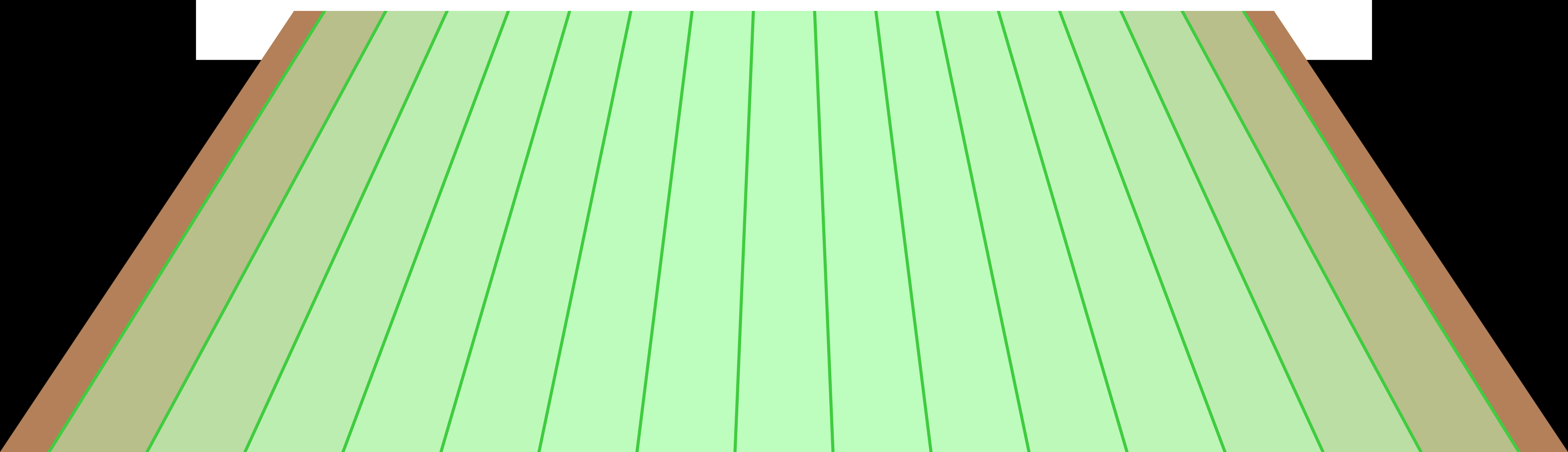
RANDOMISED ONLINE

- Lower bound construction
  - Δ partially overlapping intervals
- Adversary strategy



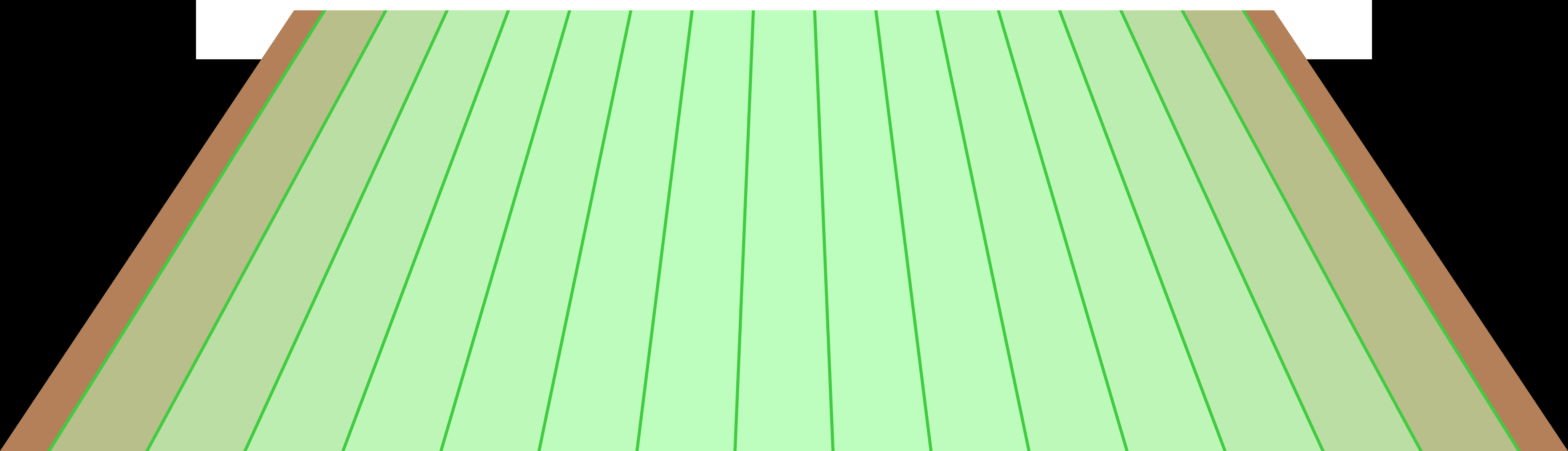
RANDOMISED ONLINE

- Lower bound construction
  - Δ partially overlapping intervals
- Adversary strategy
  - Doesn't know random choices of algorithm



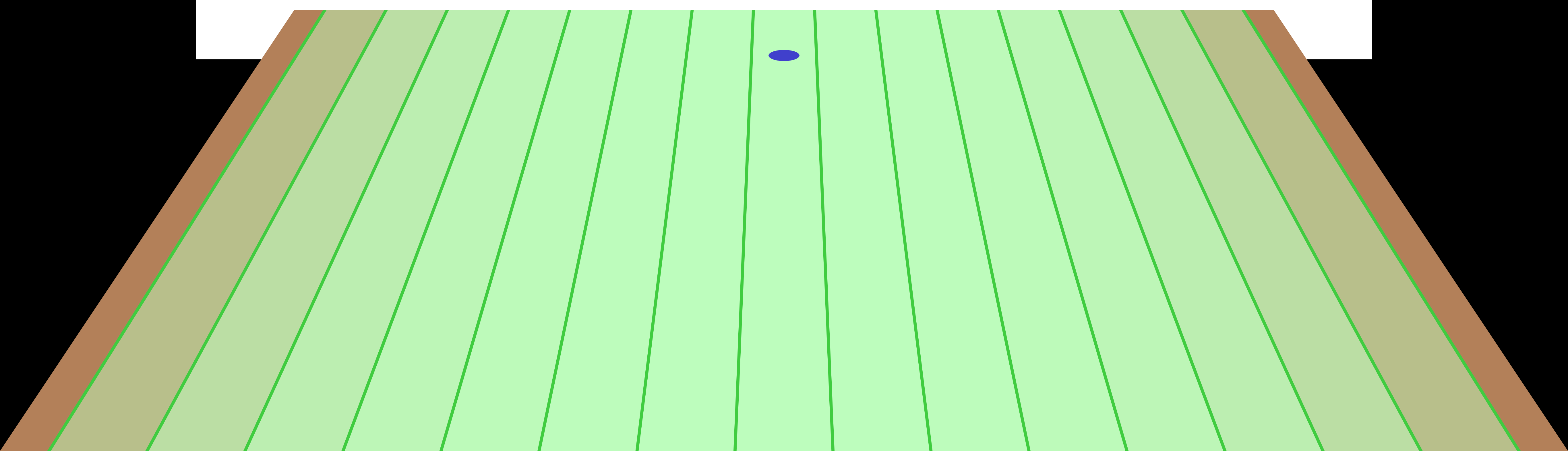
RANDOMISED ONLINE

- Lower bound construction
  - Δ partially overlapping intervals
- Adversary strategy
  - Doesn't know random choices of algorithm
  - Create tree of possible paths with high average number of handovers



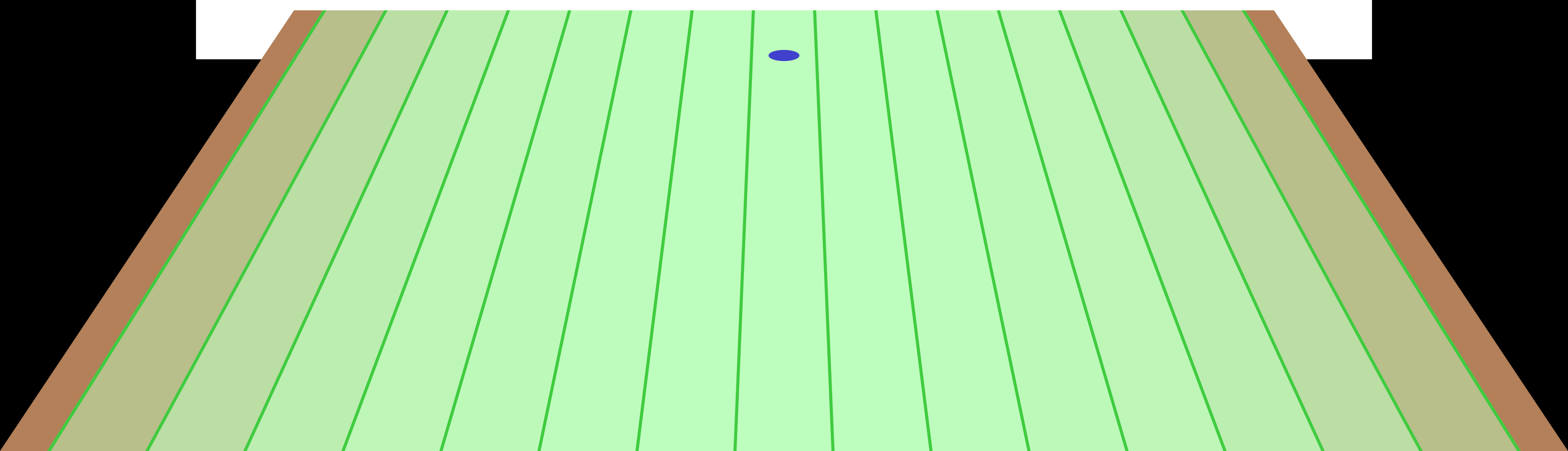
RANDOMISED ONLINE

- Lower bound construction
  - Δ partially overlapping intervals
- Adversary strategy
  - Doesn't know random choices of algorithm
  - Create tree of possible paths with high average number of handovers



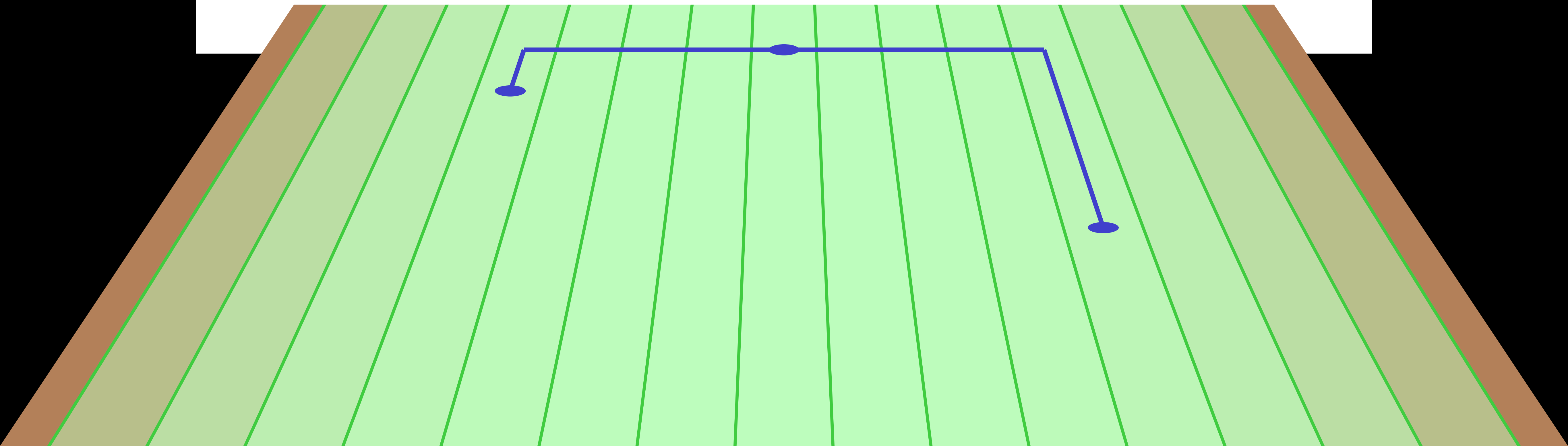
RANDOMISED ONLINE

- Lower bound construction
  - Δ partially overlapping intervals
- Adversary strategy
  - Doesn't know random choices of algorithm
  - Create tree of possible paths with high average number of handovers
  - At each node, leave half of intervals



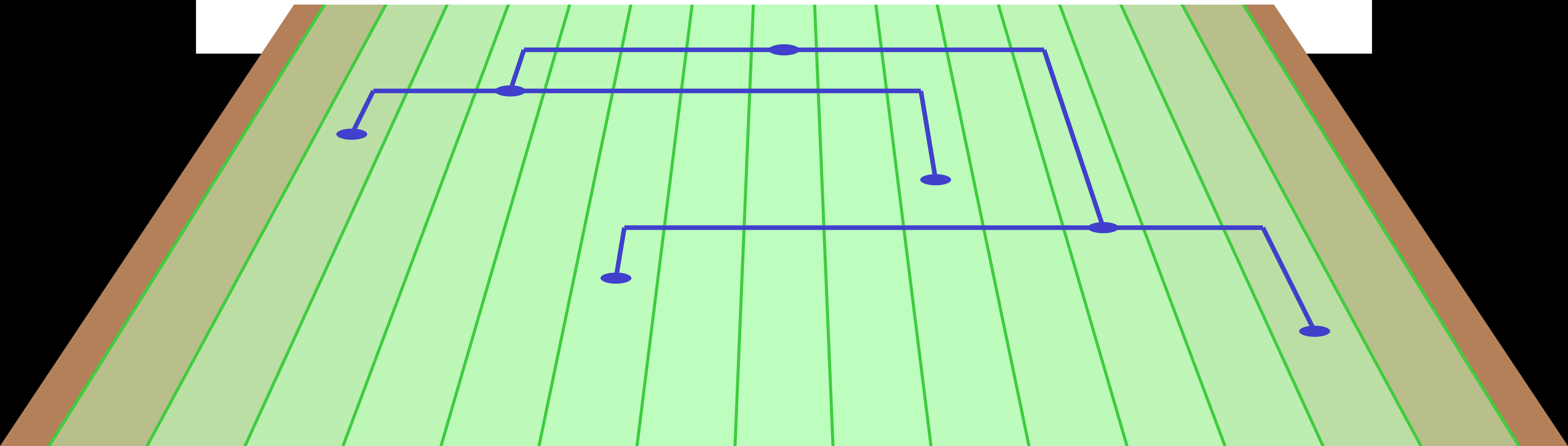
RANDOMISED ONLINE

- Lower bound construction
  - Δ partially overlapping intervals
- Adversary strategy
  - Doesn't know random choices of algorithm
  - Create tree of possible paths with high average number of handovers
  - At each node, leave half of intervals



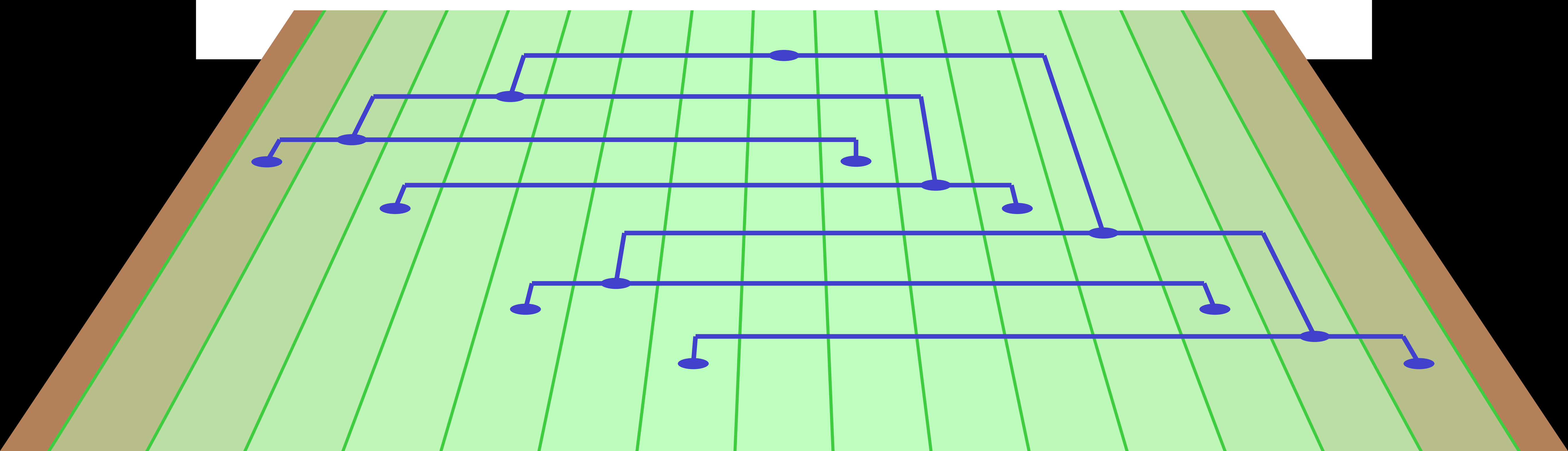
RANDOMISED ONLINE

- Lower bound construction
  - Δ partially overlapping intervals
- Adversary strategy
  - Doesn't know random choices of algorithm
  - Create tree of possible paths with high average number of handovers
  - At each node, leave half of intervals



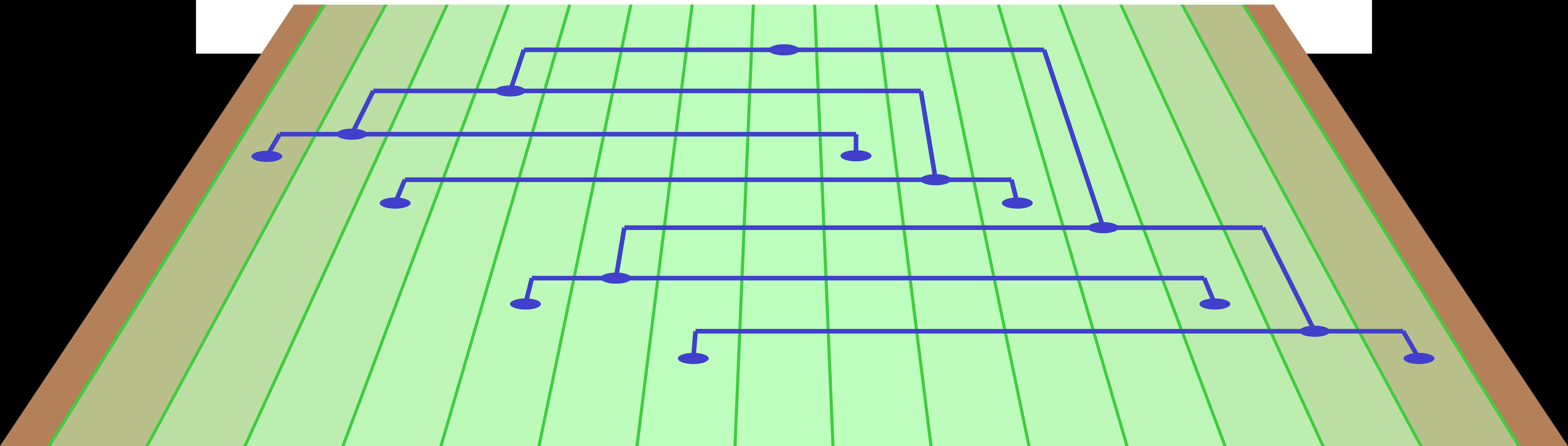
RANDOMISED ONLINE

- Lower bound construction
  - Δ partially overlapping intervals
- Adversary strategy
  - Doesn't know random choices of algorithm
  - Create tree of possible paths with high average number of handovers
  - At each node, leave half of intervals



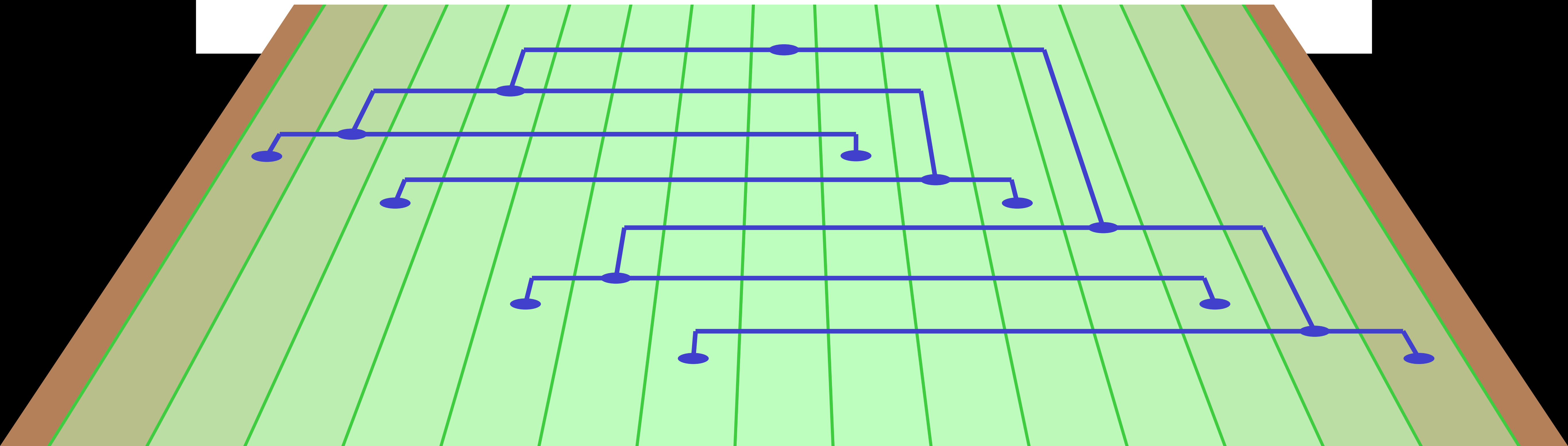
RANDOMISED ONLINE

- Lower bound construction
  - Δ partially overlapping intervals
- Adversary strategy
  - Doesn't know random choices of algorithm
  - Create tree of possible paths with high average number of handovers
  - At each node, leave half of intervals
- Optimal strategy



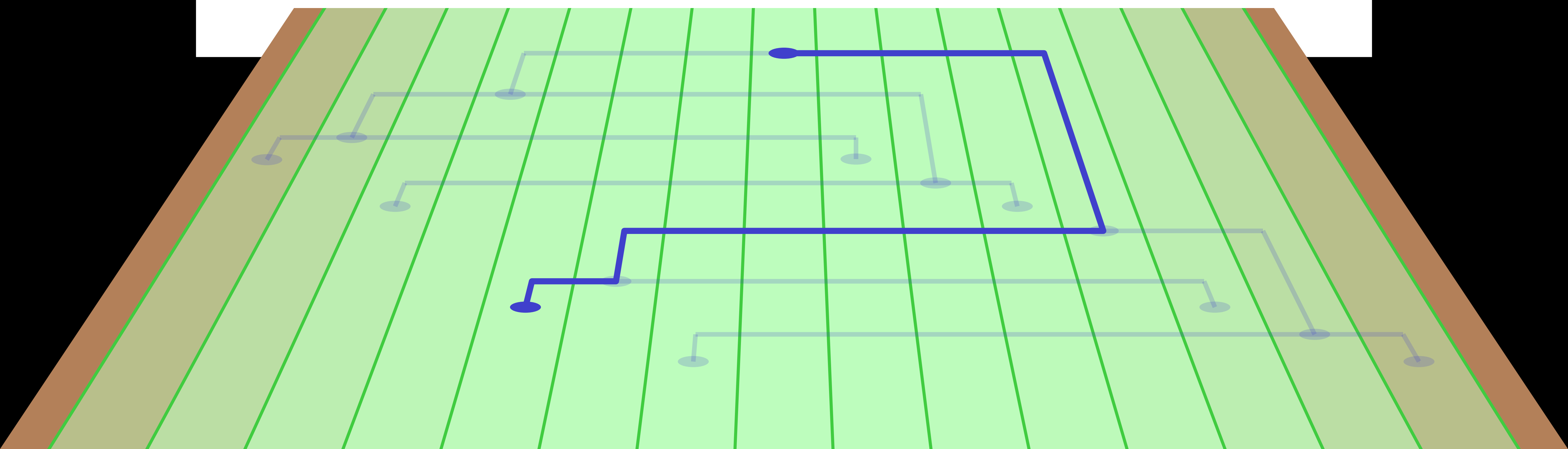
RANDOMISED ONLINE

- Lower bound construction
  - Δ partially overlapping intervals
- Adversary strategy
  - Doesn't know random choices of algorithm
  - Create tree of possible paths with high average number of handovers
  - At each node, leave half of intervals
- Optimal strategy
  - Use only one interval



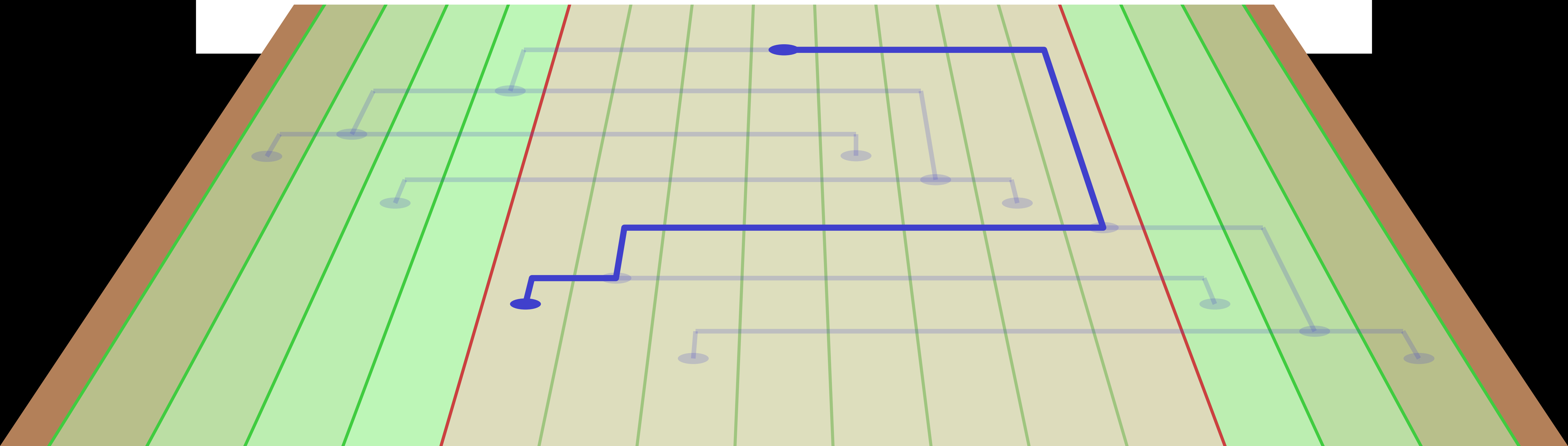
RANDOMISED ONLINE

- Lower bound construction
  - Δ partially overlapping intervals
- Adversary strategy
  - Doesn't know random choices of algorithm
  - Create tree of possible paths with high average number of handovers
  - At each node, leave half of intervals
- Optimal strategy
  - Use only one interval



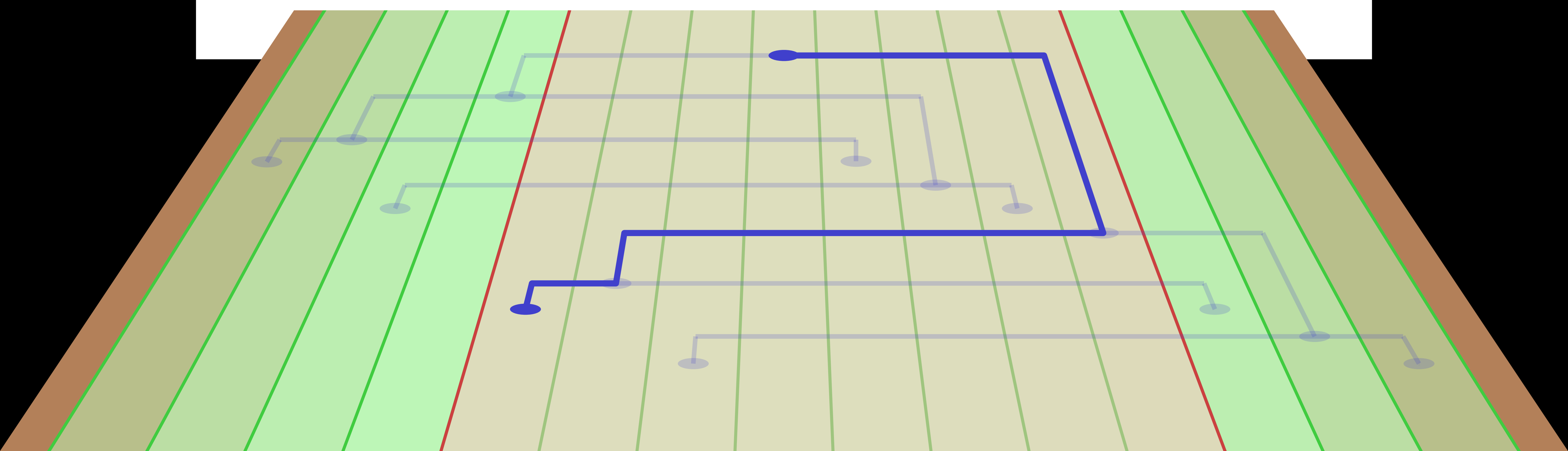
RANDOMISED ONLINE

- Lower bound construction
  - Δ partially overlapping intervals
- Adversary strategy
  - Doesn't know random choices of algorithm
  - Create tree of possible paths with high average number of handovers
  - At each node, leave half of intervals
- Optimal strategy
  - Use only one interval



RANDOMISED ONLINE

- Lower bound construction
  - Δ partially overlapping intervals
- Adversary strategy
  - Doesn't know random choices of algorithm
  - Create tree of possible paths with high average number of handovers
  - At each node, leave half of intervals
- Optimal strategy
  - Use only one interval
  - Competitive ratio $\log \Delta$



CONCLUSIONS

CONCLUSIONS

- This work

CONCLUSIONS

- This work
  - Matching bounds on competitive ratio in many settings

CONCLUSIONS

- This work
  - Matching bounds on competitive ratio in many settings
  - Works for c-lateration

CONCLUSIONS

- This work
  - Matching bounds on competitive ratio in many settings
  - Works for c-lateration
  - Works for multiple independent UMOs

CONCLUSIONS

- This work
  - Matching bounds on competitive ratio in many settings
  - Works for c-lateration
  - Works for multiple independent UMOs
- Future work

CONCLUSIONS

- This work
  - Matching bounds on competitive ratio in many settings
  - Works for c-lateration
  - Works for multiple independent UMOs
- Future work
  - Multiple UMOs with capacity limit: each sensor can track at most k UMOs at a time

CONCLUSIONS

- This work
  - Matching bounds on competitive ratio in many settings
  - Works for c-lateration
  - Works for multiple independent UMOs
- Future work
  - Multiple UMOs with capacity limit: each sensor can track at most k UMOs at a time

THANK YOU!