Towards the Minimization of Global Measures of Congestion Potential for Moving Points

Will Evans
Ivor van der Hoog
David Kirkpatrick
Maarten Löffler

MOVING POINTS

MOVING POINTS

Location-aware mobile devices

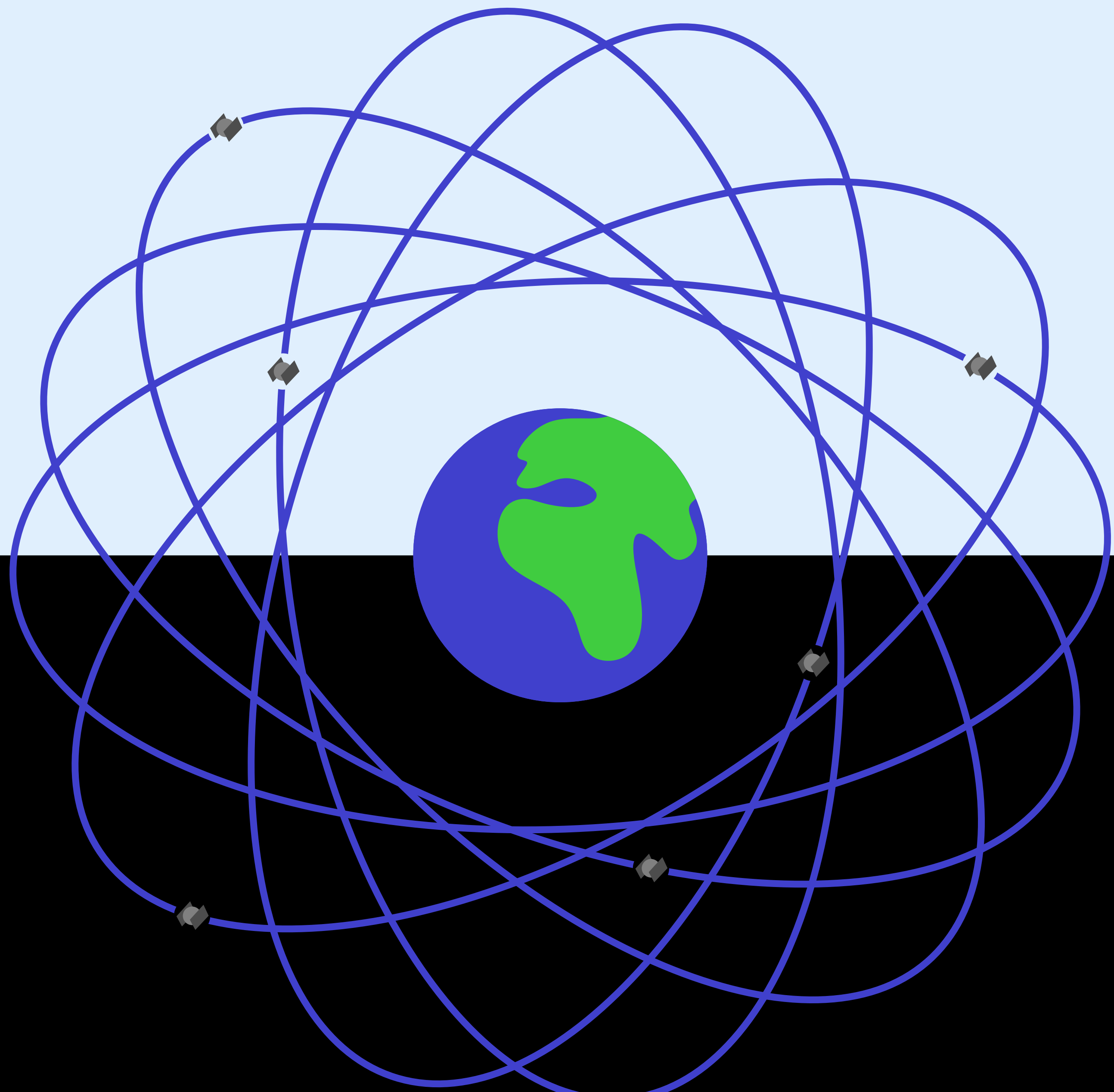
MOVING POINTS

- Location-aware mobile devices
  - GPS systems

MOVING POINTS

Location-aware mobile devices

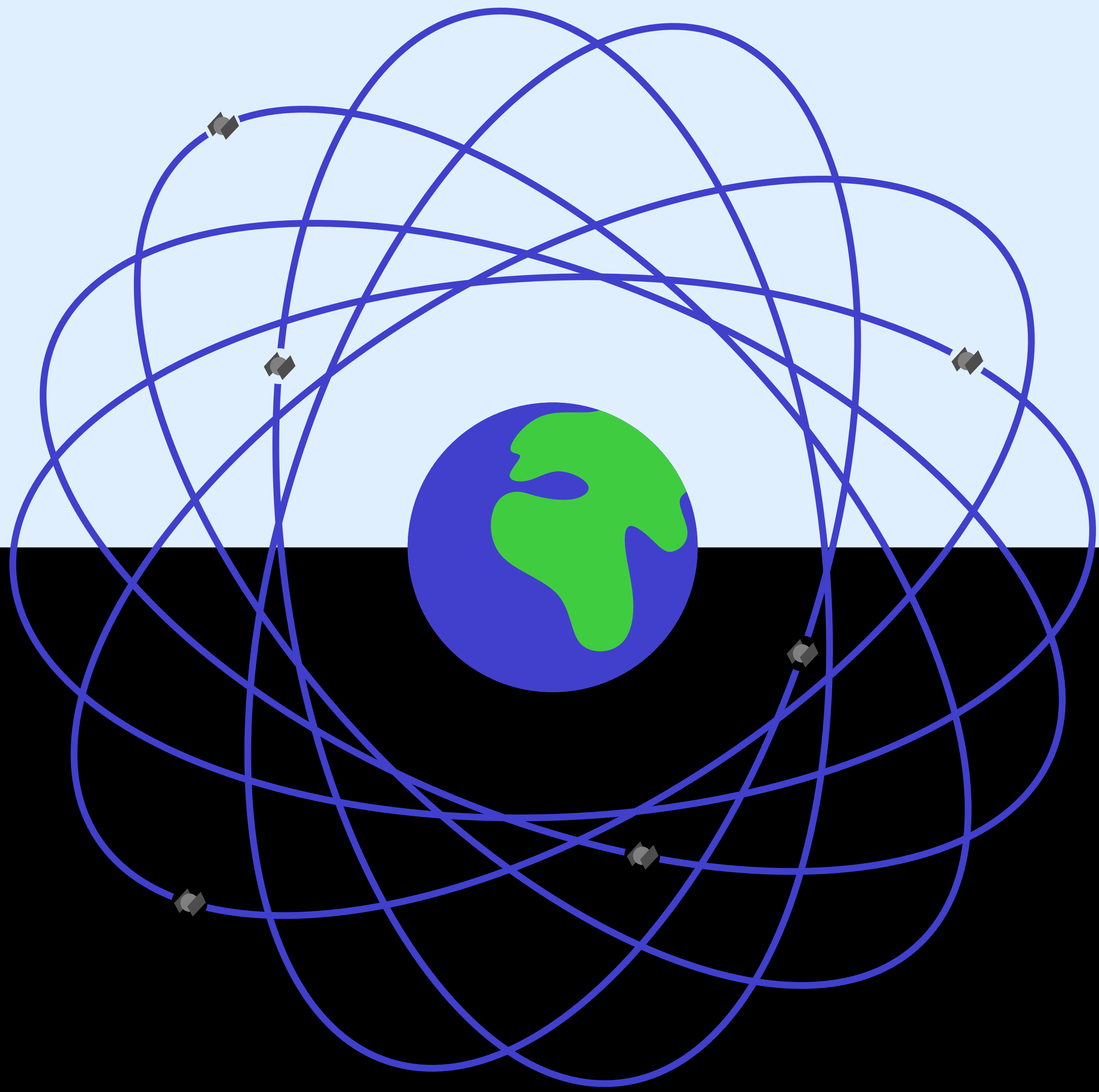
GPS systems



MOVING POINTS

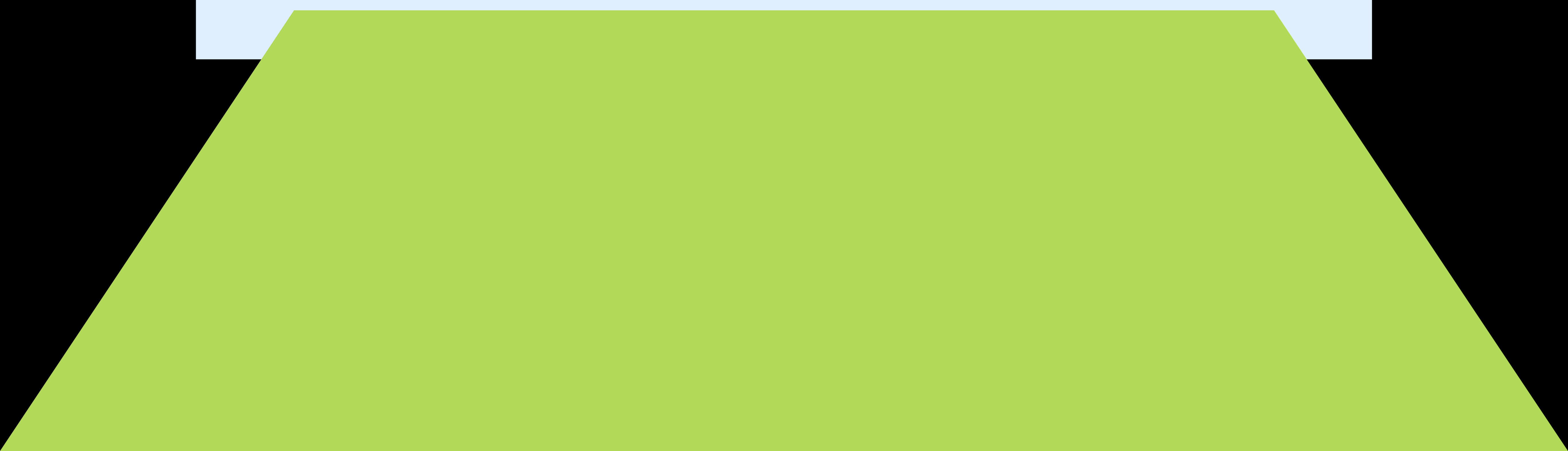
Location-aware mobile devices

- GPS systems
- Smart phones



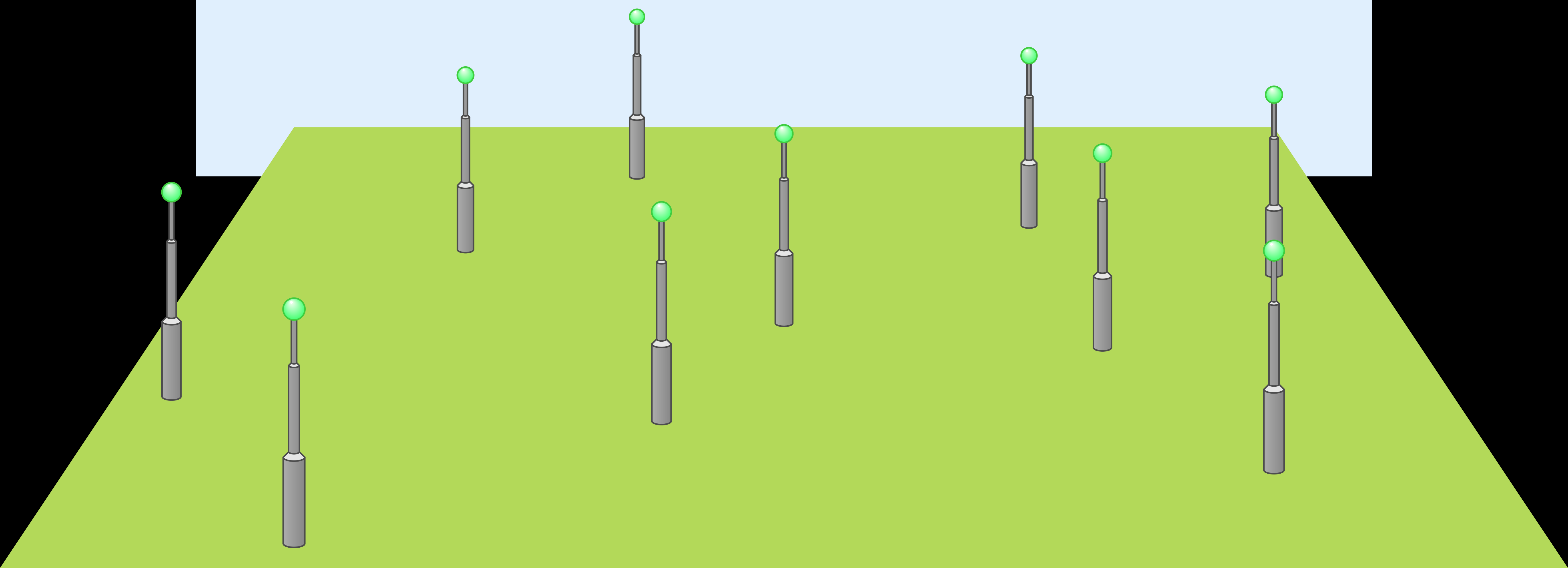
MOVING POINTS

- Location-aware mobile devices
  - GPS systems
  - Smart phones



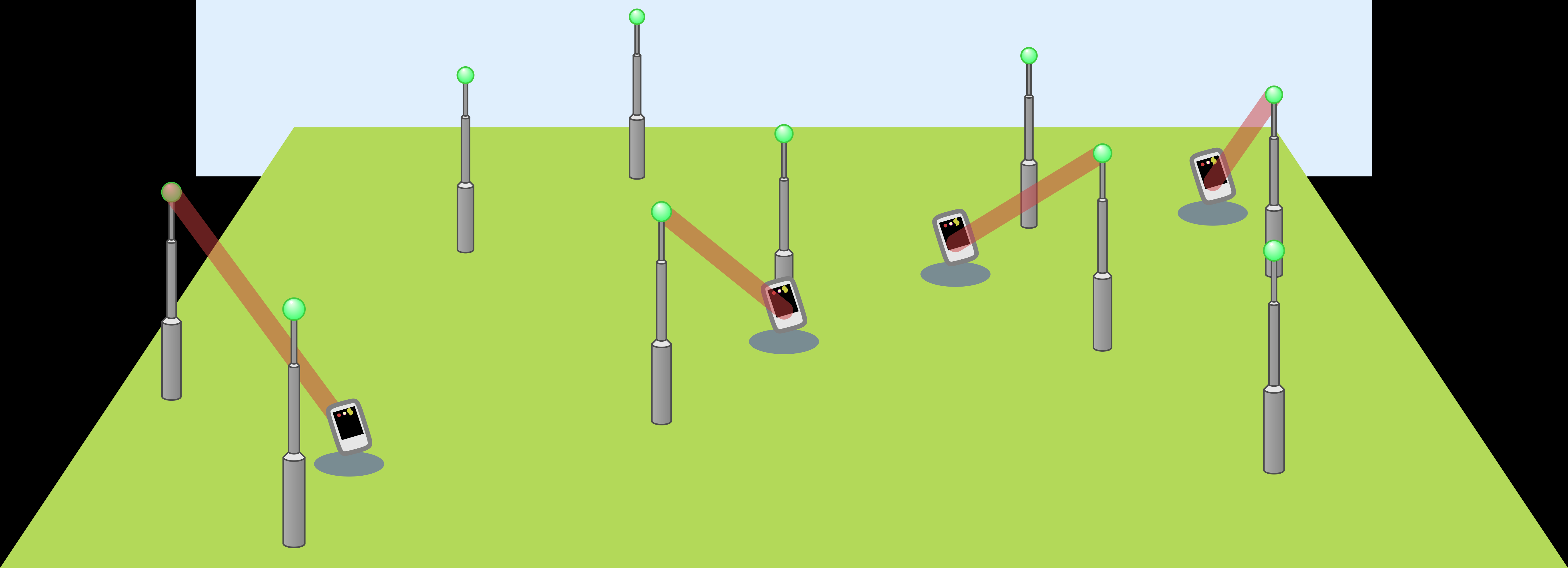
MOVING POINTS

- Location-aware mobile devices
  - GPS systems
  - Smart phones



MOVING POINTS

- Location-aware mobile devices
  - GPS systems
  - Smart phones

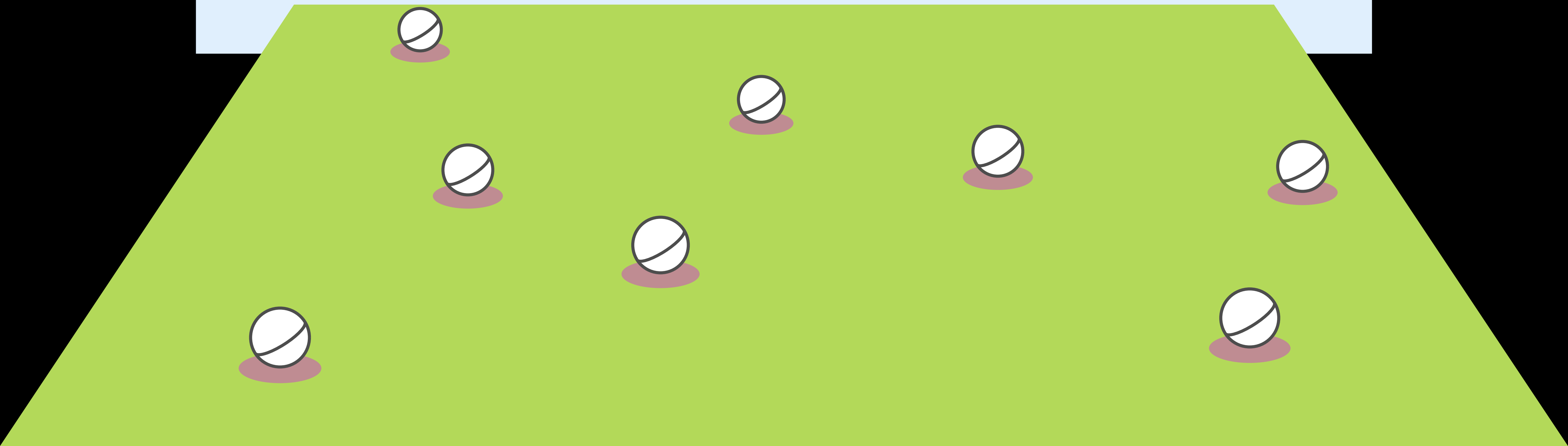


MOVING POINTS

- Location-aware mobile devices
  - GPS systems
  - Smart phones
  - Wireless sensor networks

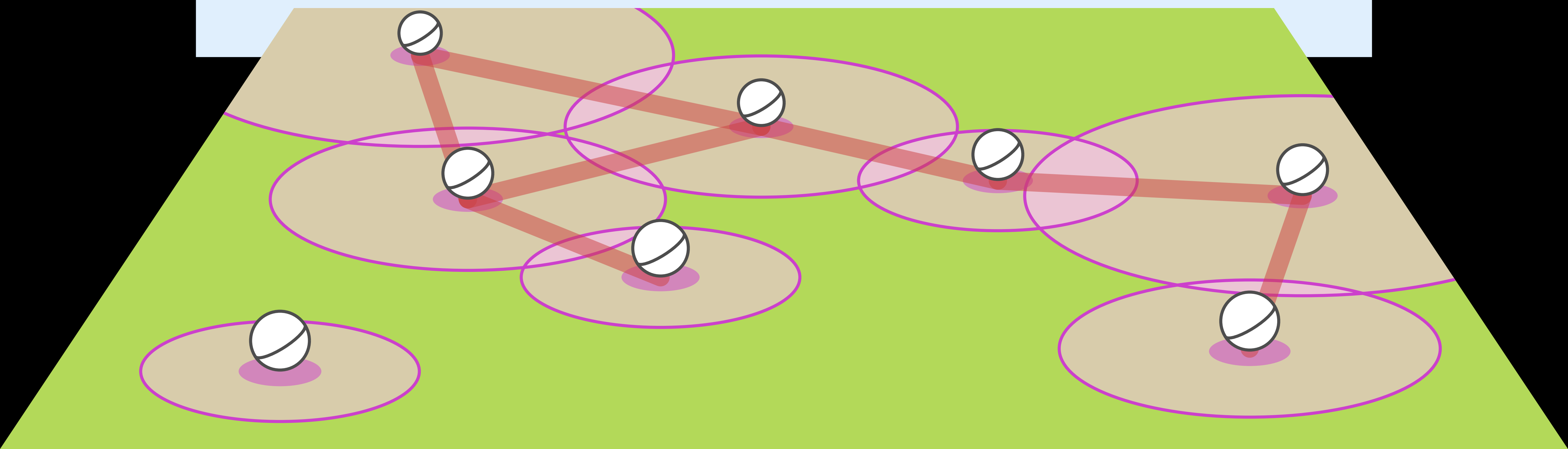
MOVING POINTS

- Location-aware mobile devices
  - GPS systems
  - Smart phones
  - Wireless sensor networks



MOVING POINTS

- Location-aware mobile devices
  - GPS systems
  - Smart phones
  - Wireless sensor networks



MOVING POINTS

- Location-aware mobile devices
  - GPS systems
  - Smart phones
  - Wireless sensor networks
  - Flock tracking

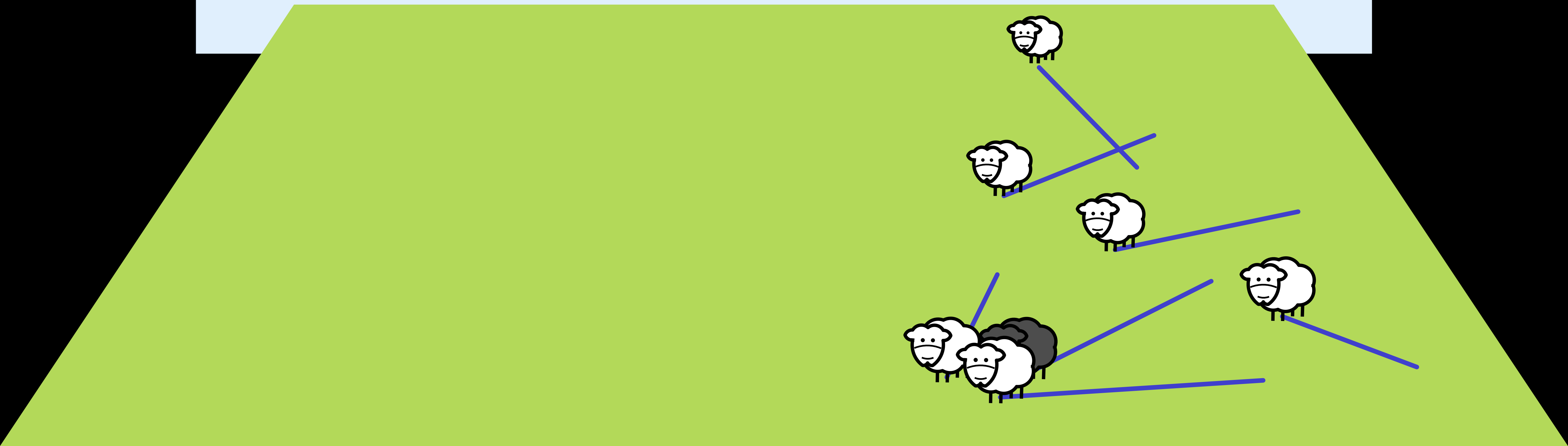
MOVING POINTS

- Location-aware mobile devices
  - GPS systems
  - Smart phones
  - Wireless sensor networks
  - Flock tracking

MOVING POINTS

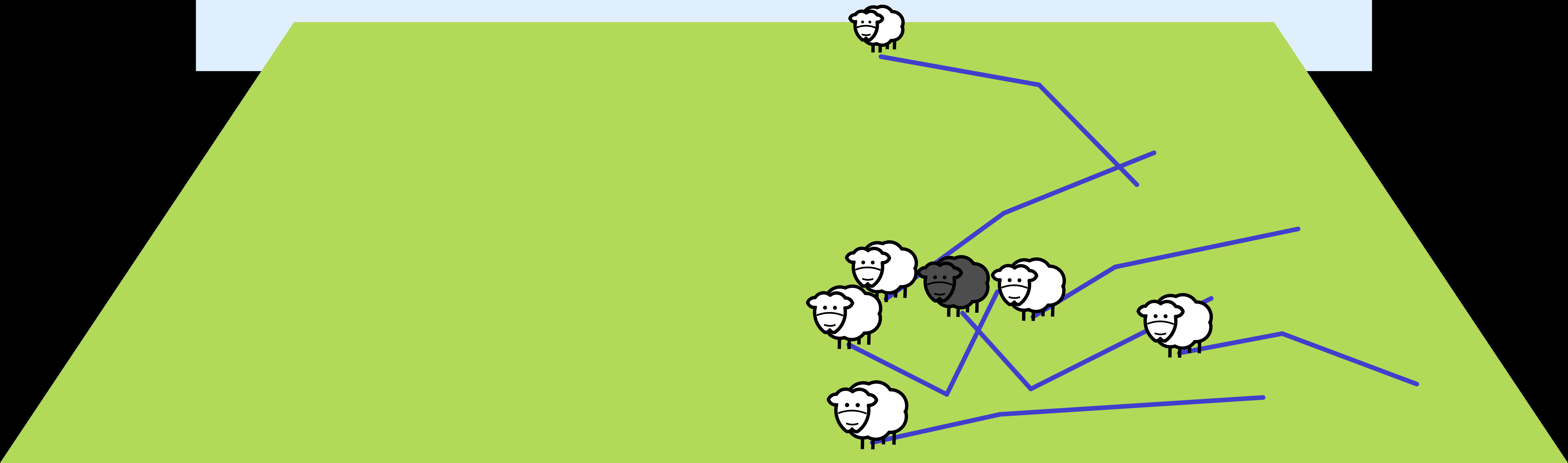
Location-aware mobile devices

- GPS systems
- Smart phones
- Wireless sensor networks
- Flock tracking



MOVING POINTS

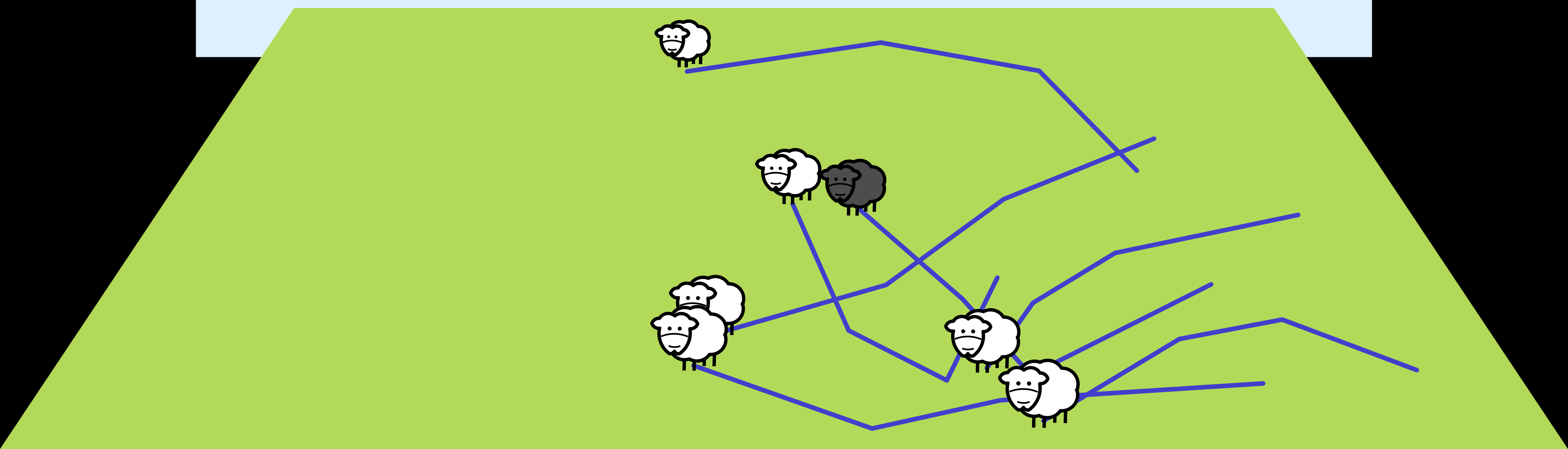
- Location-aware mobile devices
  - GPS systems
  - Smart phones
  - Wireless sensor networks
  - Flock tracking



MOVING POINTS

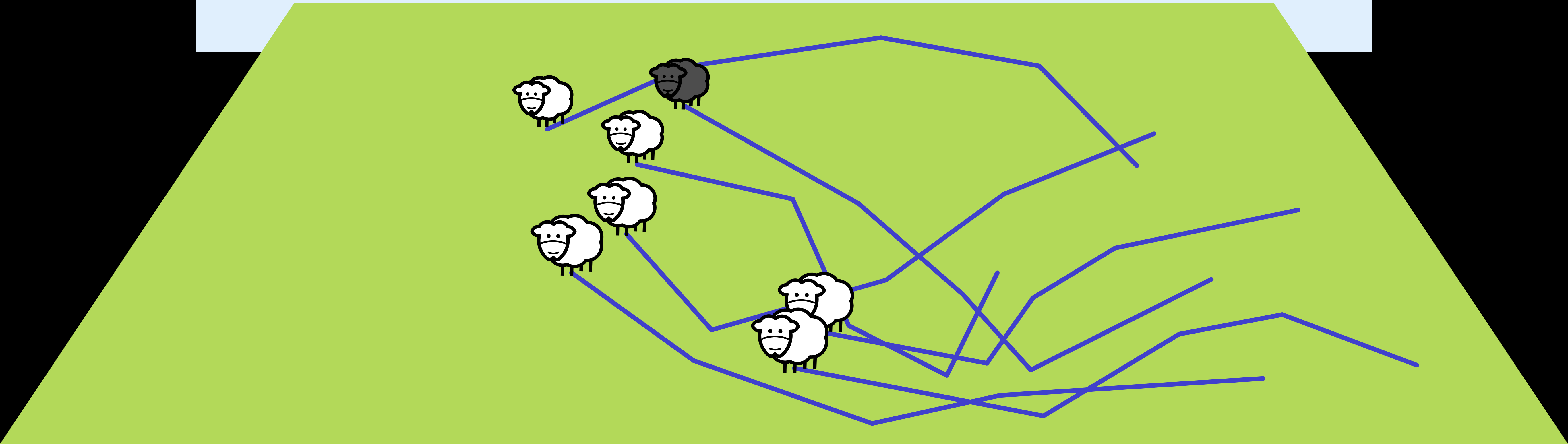
Location-aware mobile devices

- GPS systems
- Smart phones
- Wireless sensor networks
- Flock tracking



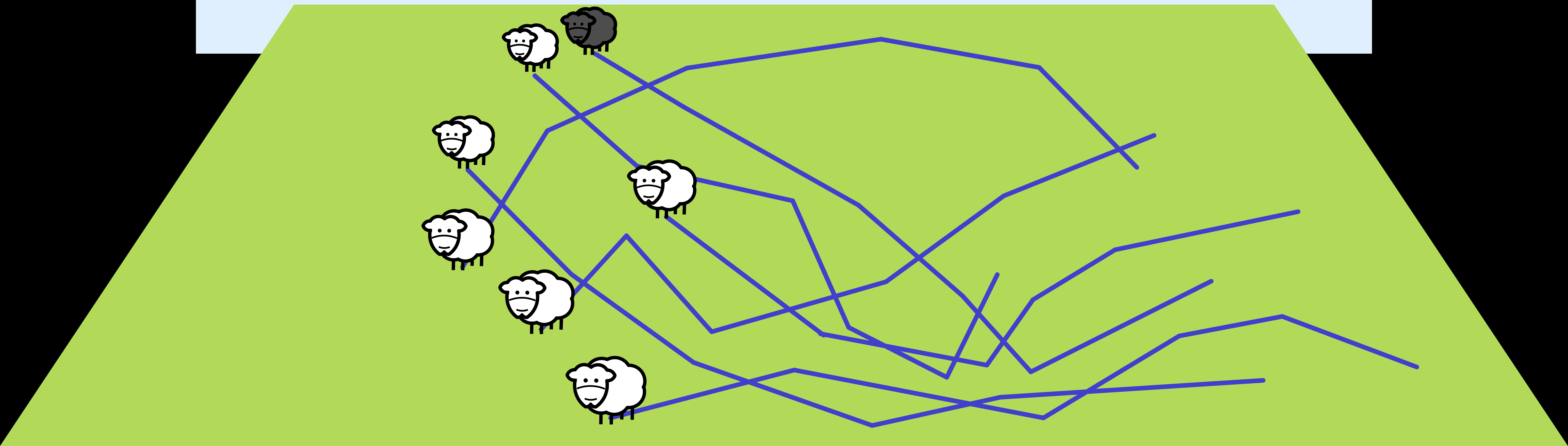
MOVING POINTS

- Location-aware mobile devices
  - GPS systems
  - Smart phones
  - Wireless sensor networks
  - Flock tracking



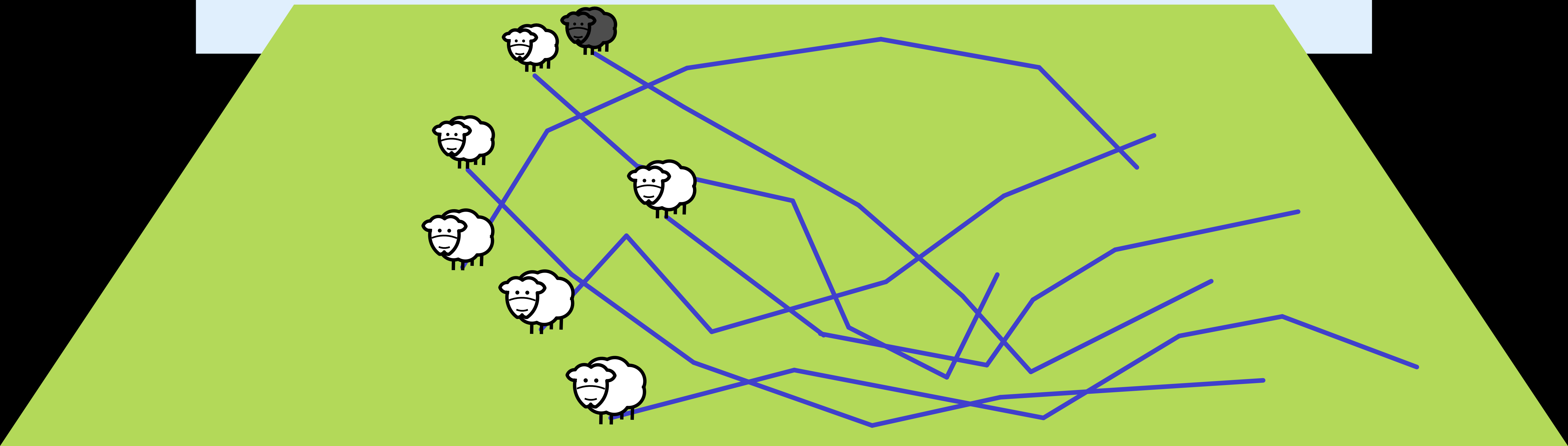
MOVING POINTS

- Location-aware mobile devices
  - GPS systems
  - Smart phones
  - Wireless sensor networks
  - Flock tracking



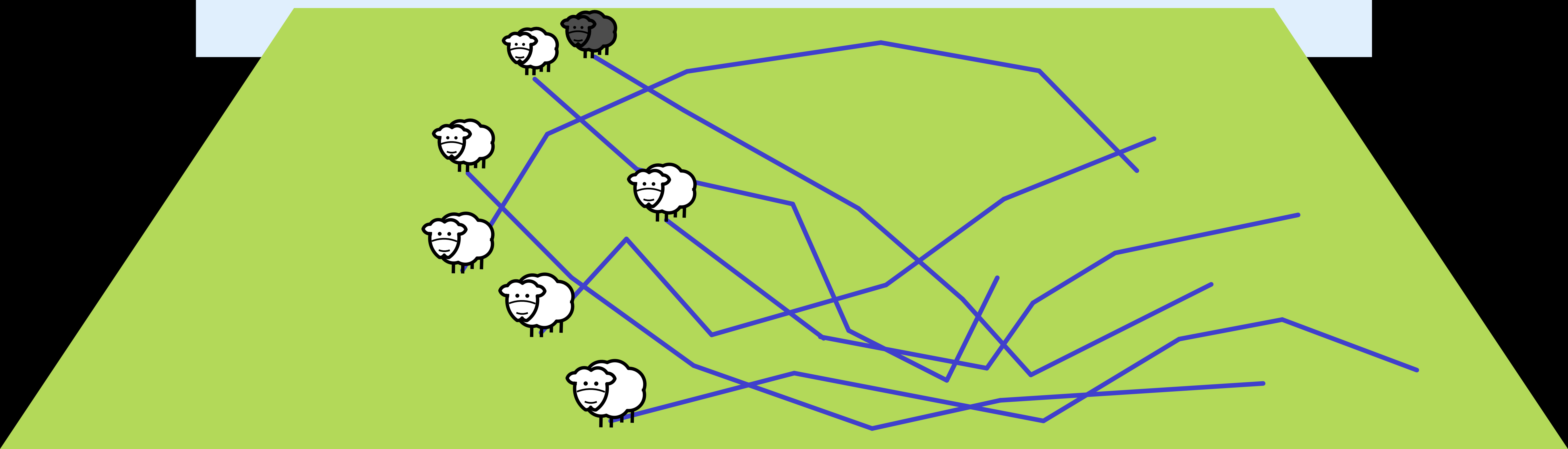
MOVING POINTS

- Location-aware mobile devices
  - GPS systems
  - Smart phones
  - Wireless sensor networks
  - Flock tracking
- New geometric data sets



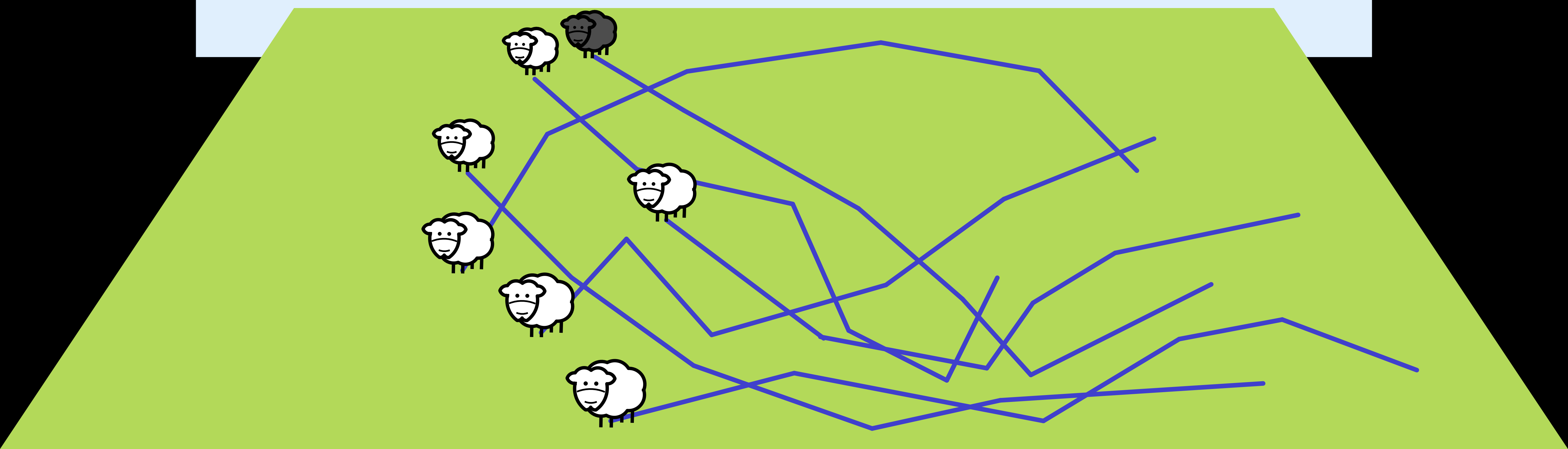
MOVING POINTS

- Location-aware mobile devices
  - GPS systems
  - Smart phones
  - Wireless sensor networks
  - Flock tracking
- New geometric data sets
  - Often very large



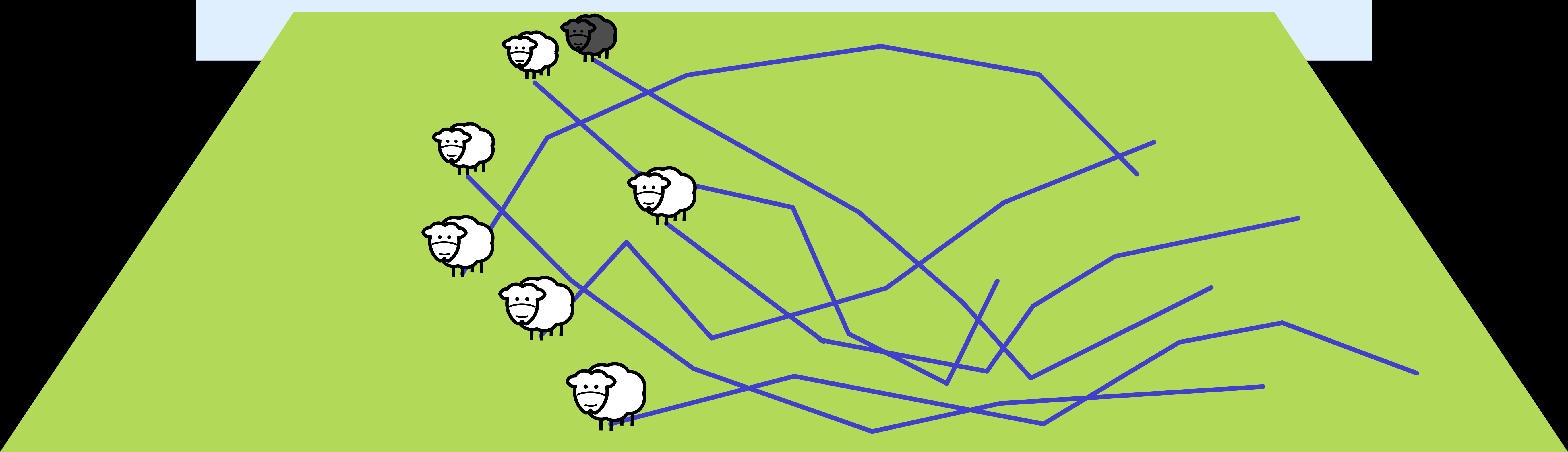
MOVING POINTS

- Location-aware mobile devices
  - GPS systems
  - Smart phones
  - Wireless sensor networks
  - Flock tracking
- New geometric data sets
  - Often very large
  - Imprecise in nature



MOVING POINTS

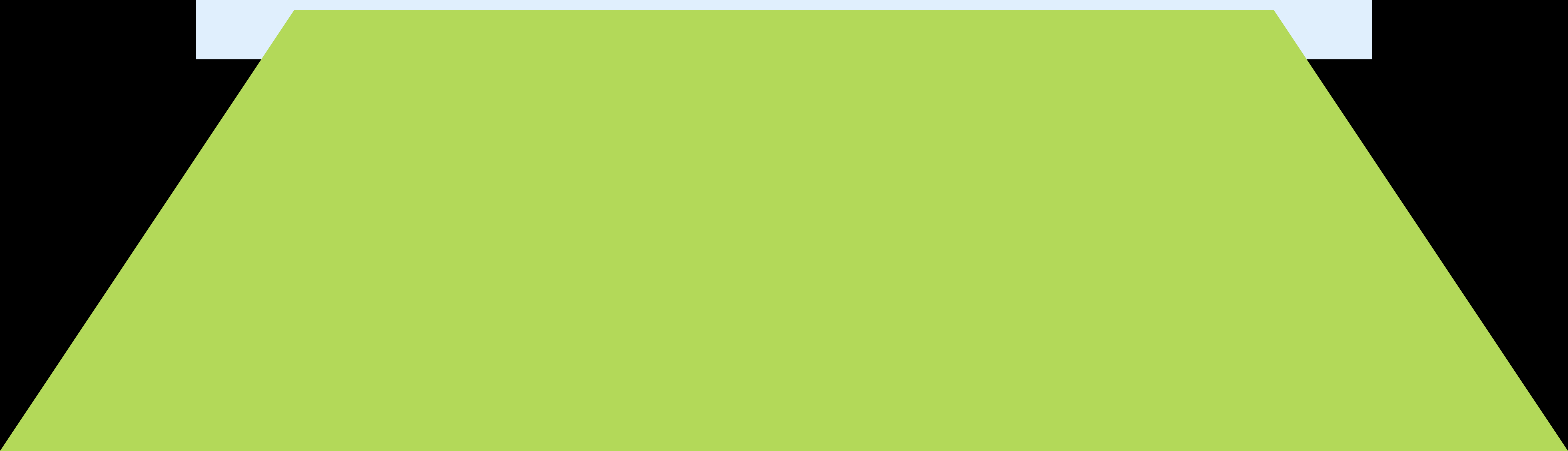
- Location-aware mobile devices
  - GPS systems
  - Smart phones
  - Wireless sensor networks
  - Flock tracking
- New geometric data sets
  - Often very large
  - Imprecise in nature
  - Dynamic / unpredictable



UNCERTAIN POINTS

UNCERTAIN POINTS

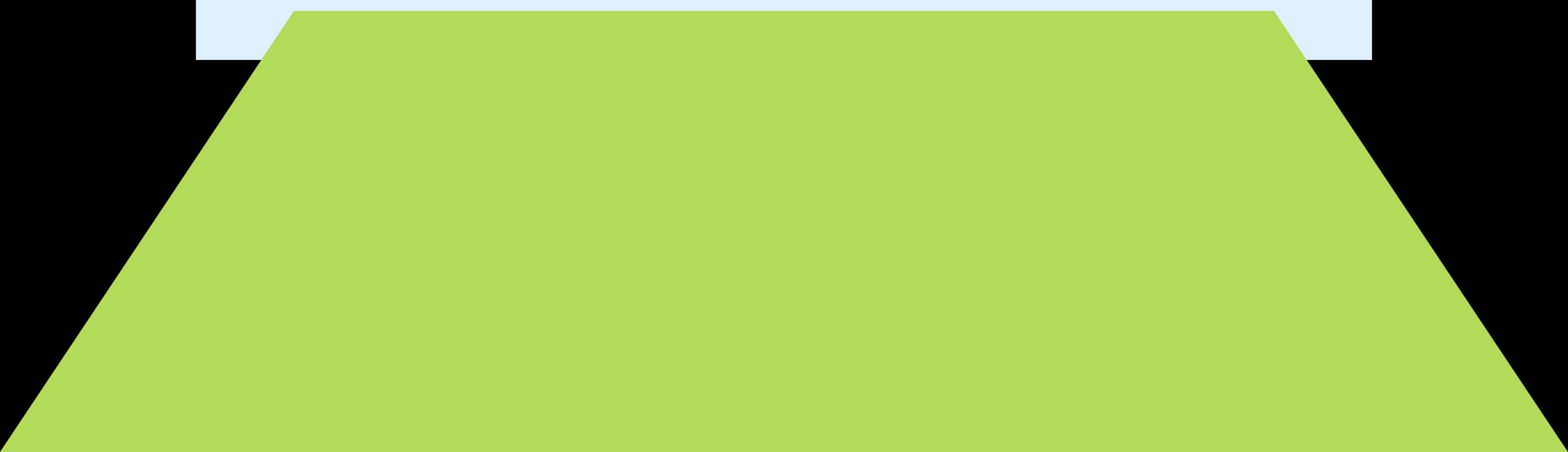
Motion causes imprecision



UNCERTAIN POINTS

Motion causes imprecision

Suppose we have a moving point p



UNCERTAIN POINTS

Motion causes imprecision

Suppose we have a moving point p

UNCERTAIN POINTS

Motion causes imprecision

- Suppose we have a moving point p
- We know an upper bound on its speed

UNCERTAIN POINTS

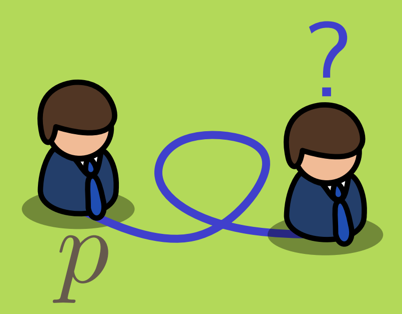
Motion causes imprecision

- Suppose we have a moving point p
- We know an upper bound on its speed
- In 1 time step p can move at most d

UNCERTAIN POINTS

Motion causes imprecision

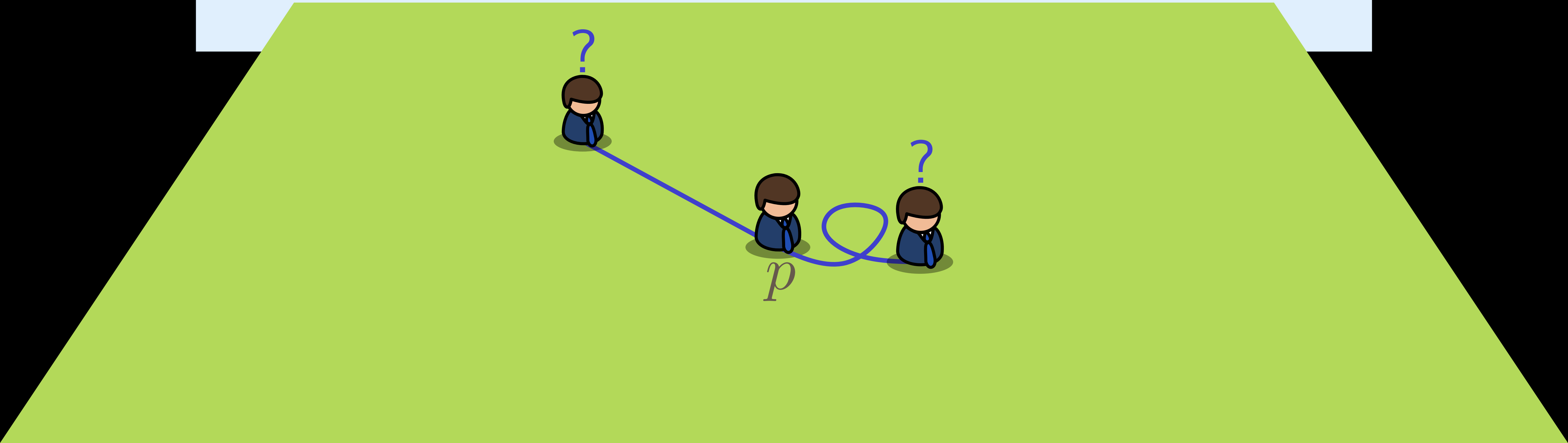
- Suppose we have a moving point p
- We know an upper bound on its speed
- In 1 time step p can move at most d



UNCERTAIN POINTS

Motion causes imprecision

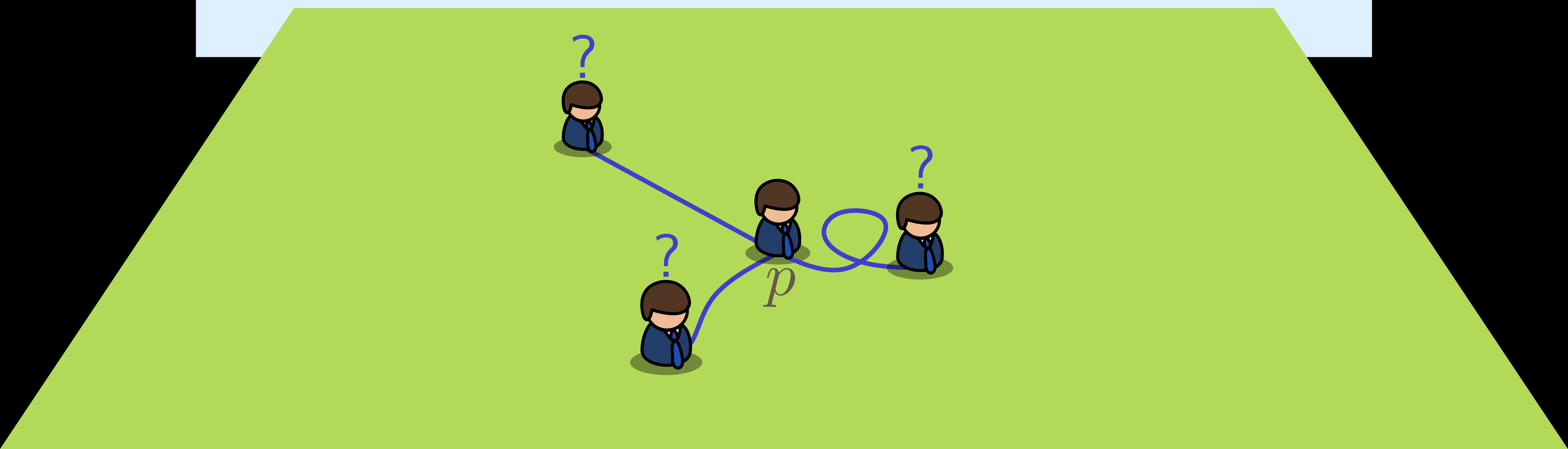
- Suppose we have a moving point p
- We know an upper bound on its speed
- In 1 time step p can move at most d



UNCERTAIN POINTS

Motion causes imprecision

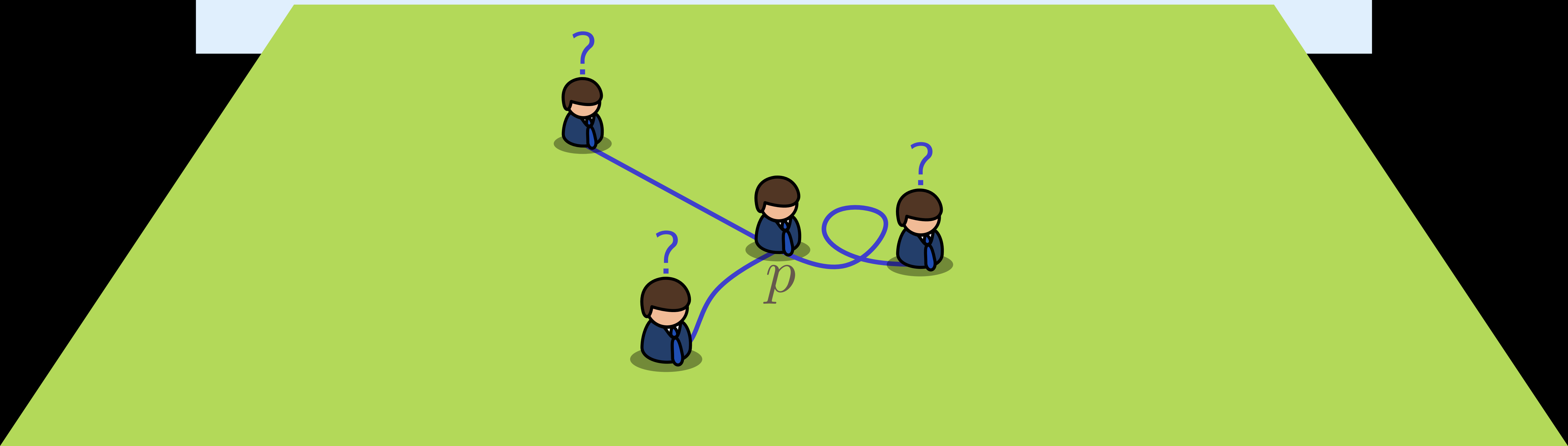
- Suppose we have a moving point p
- We know an upper bound on its speed
- In 1 time step p can move at most d



UNCERTAIN POINTS

Motion causes imprecision

- Suppose we have a moving point p
- We know an upper bound on its speed
- In 1 time step p can move at most d
- We model the *uncertainty region* of p in the next time step by a disk of radius d



UNCERTAIN POINTS

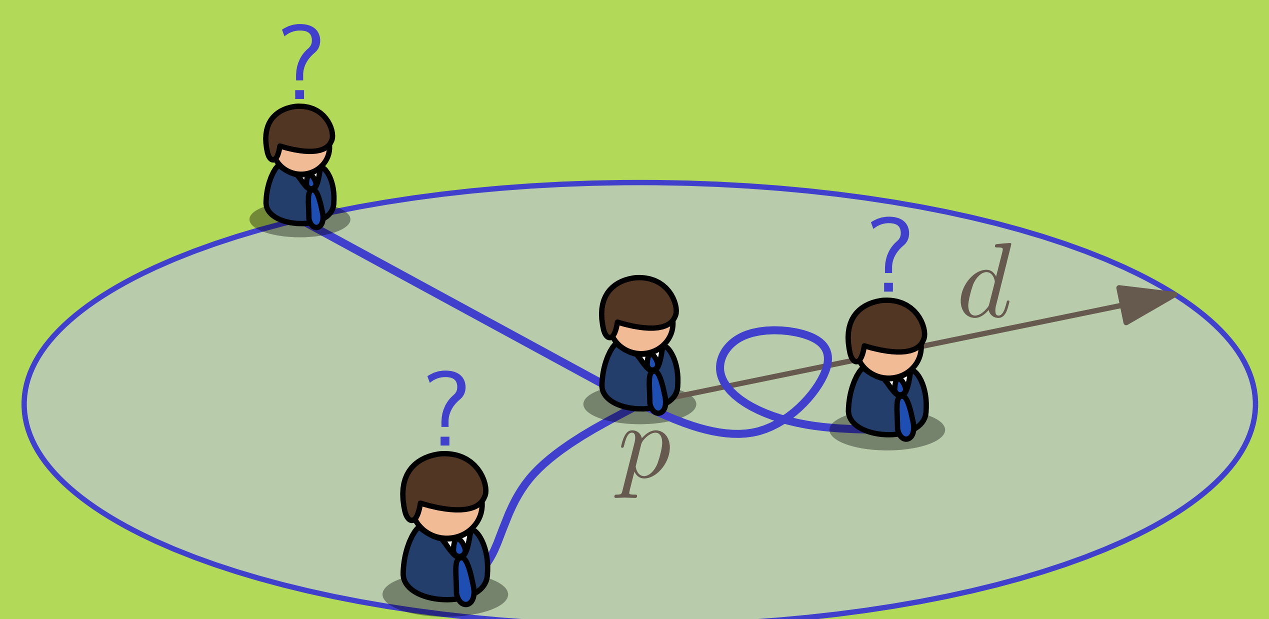
Motion causes imprecision

Suppose we have a moving point p

We know an upper bound on its speed

In 1 time step p can move at most d

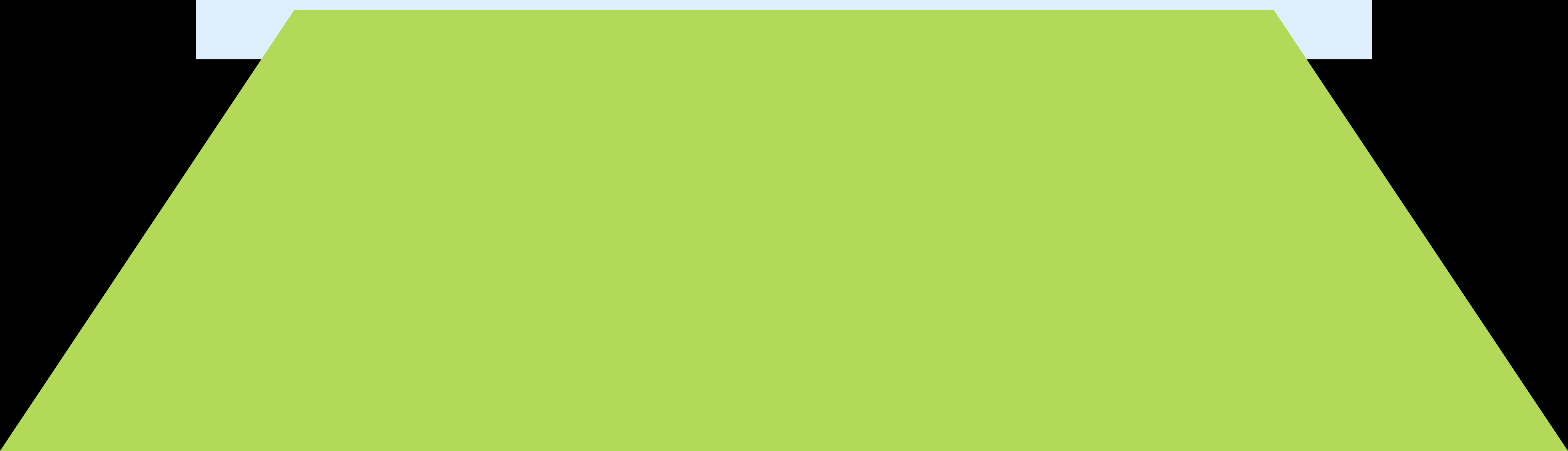
We model the *uncertainty region* of p in the next time step by a disk of radius d



UNCERTAIN POINTS

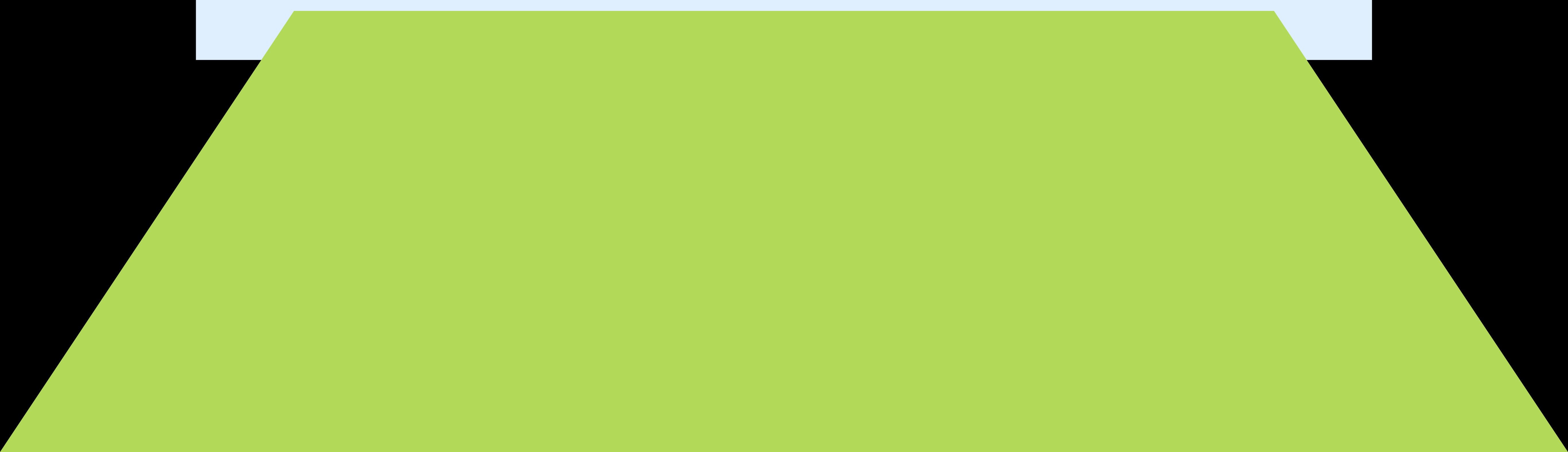
UNCERTAIN POINTS

Consider n moving points



UNCERTAIN POINTS

- Consider n moving points
  - Suppose that for each point we know a *location* p_i and an associated *time stamp* t_i

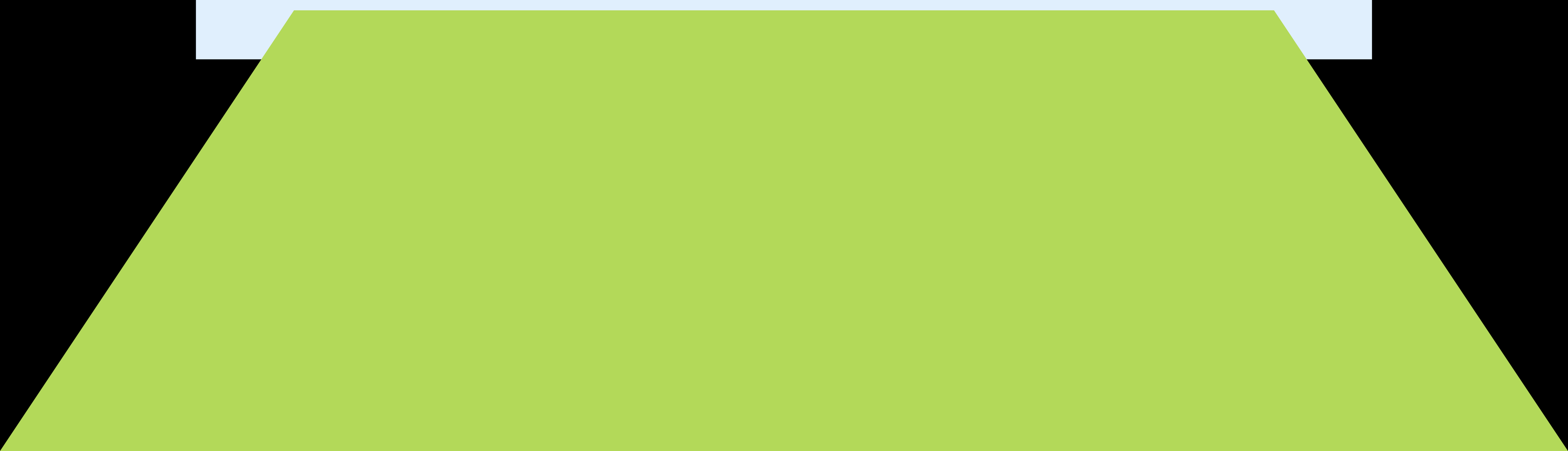


UNCERTAIN POINTS

Consider n moving points

Suppose that for each point we know a *location* p_i and an associated *time stamp* t_i

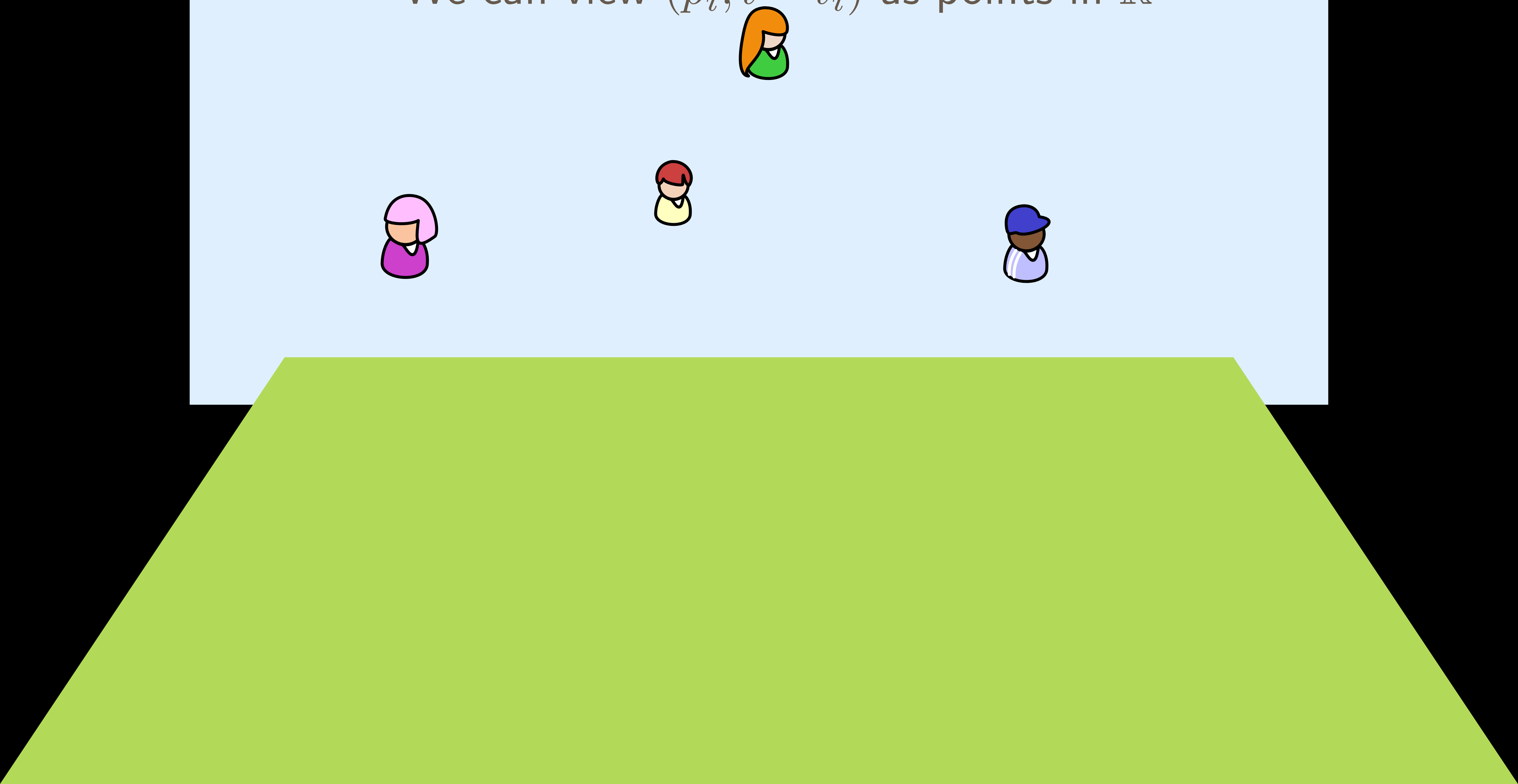
We can view $(p_i, t - t_i)$ as points in $\mathbb{R}^3$



UNCERTAIN POINTS

Consider n moving points

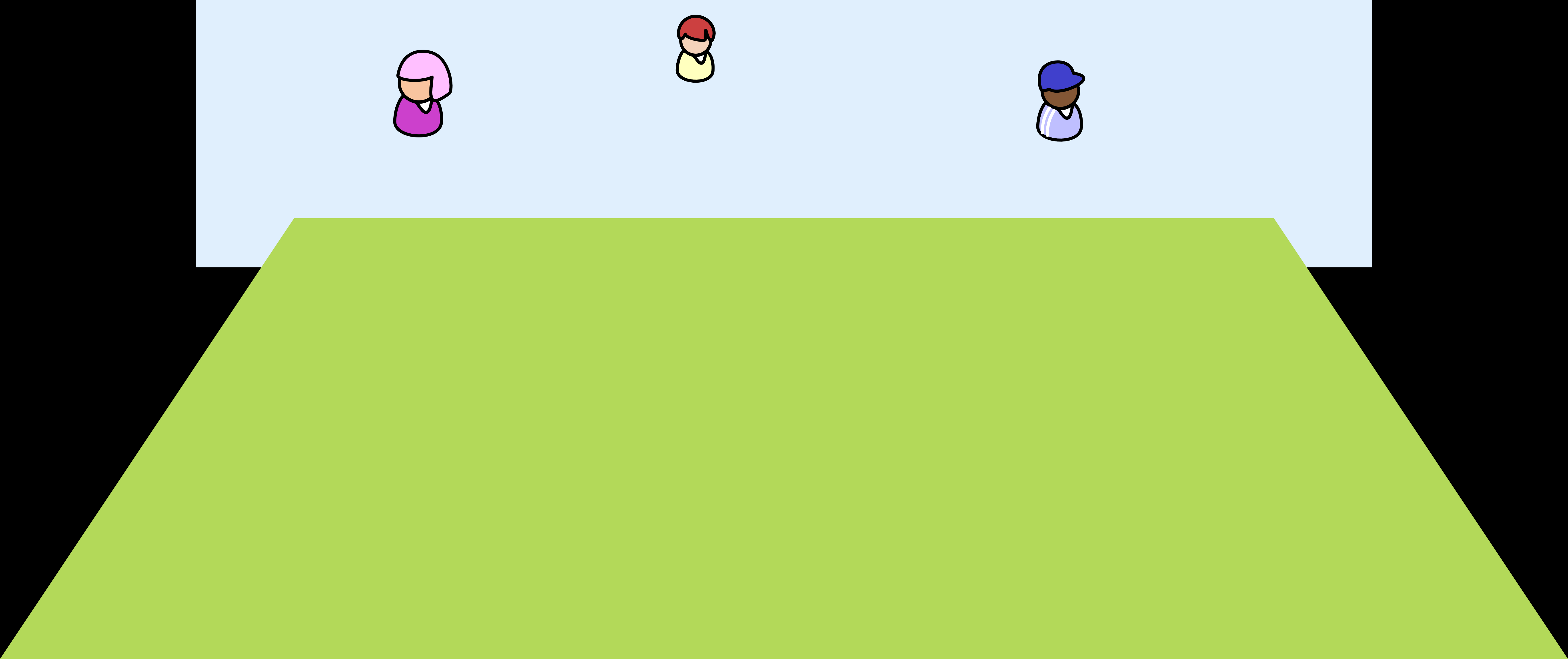
- Suppose that for each point we know a *location* p_i and an associated *time stamp* t_i
- We can view $(p_i, t - t_i)$ as points in $\mathbb{R}^3$



UNCERTAIN POINTS

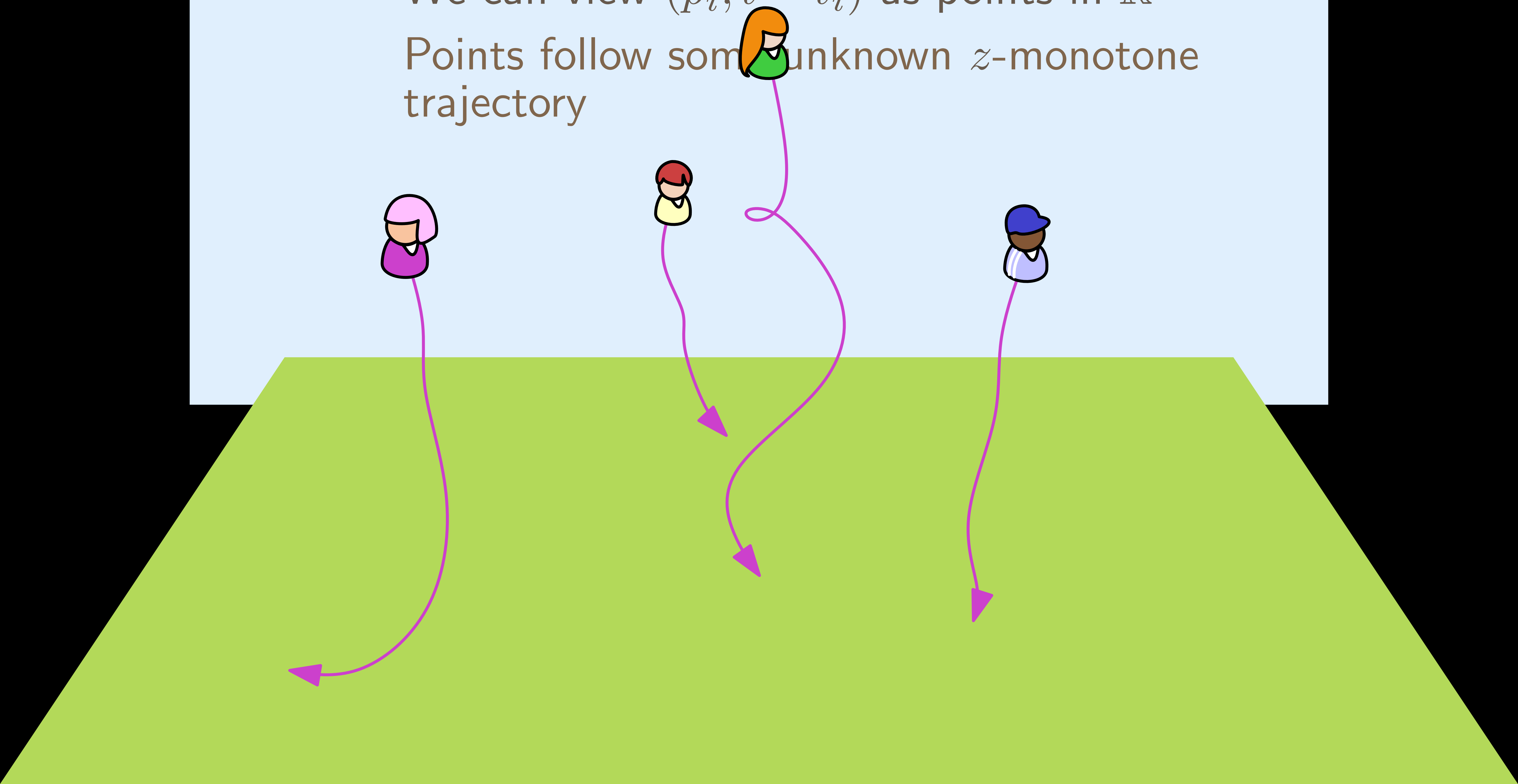
Consider n moving points

- Suppose that for each point we know a *location* p_i and an associated *time stamp* t_i
- We can view $(p_i, t - t_i)$ as points in $\mathbb{R}^3$
- Points follow some unknown z-monotone trajectory



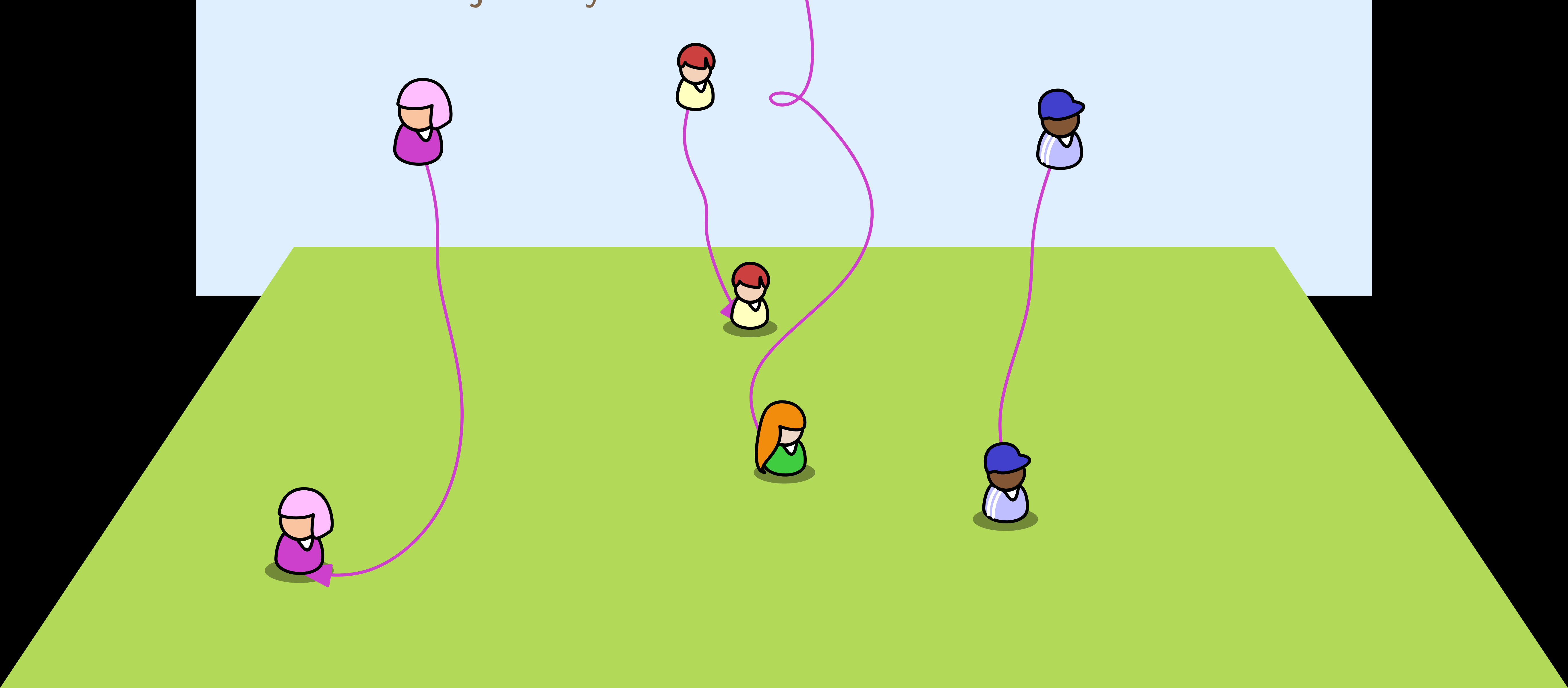
UNCERTAIN POINTS

- Consider n moving points
  - Suppose that for each point we know a *location* p_i and an associated *time stamp* t_i
  - We can view $(p_i, t - t_i)$ as points in $\mathbb{R}^3$
  - Points follow some unknown z-monotone trajectory



UNCERTAIN POINTS

- Consider n moving points
  - Suppose that for each point we know a *location* p_i and an associated *time stamp* t_i
  - We can view $(p_i, t - t_i)$ as points in $\mathbb{R}^3$
  - Points follow some unknown z-monotone trajectory



UNCERTAIN POINTS

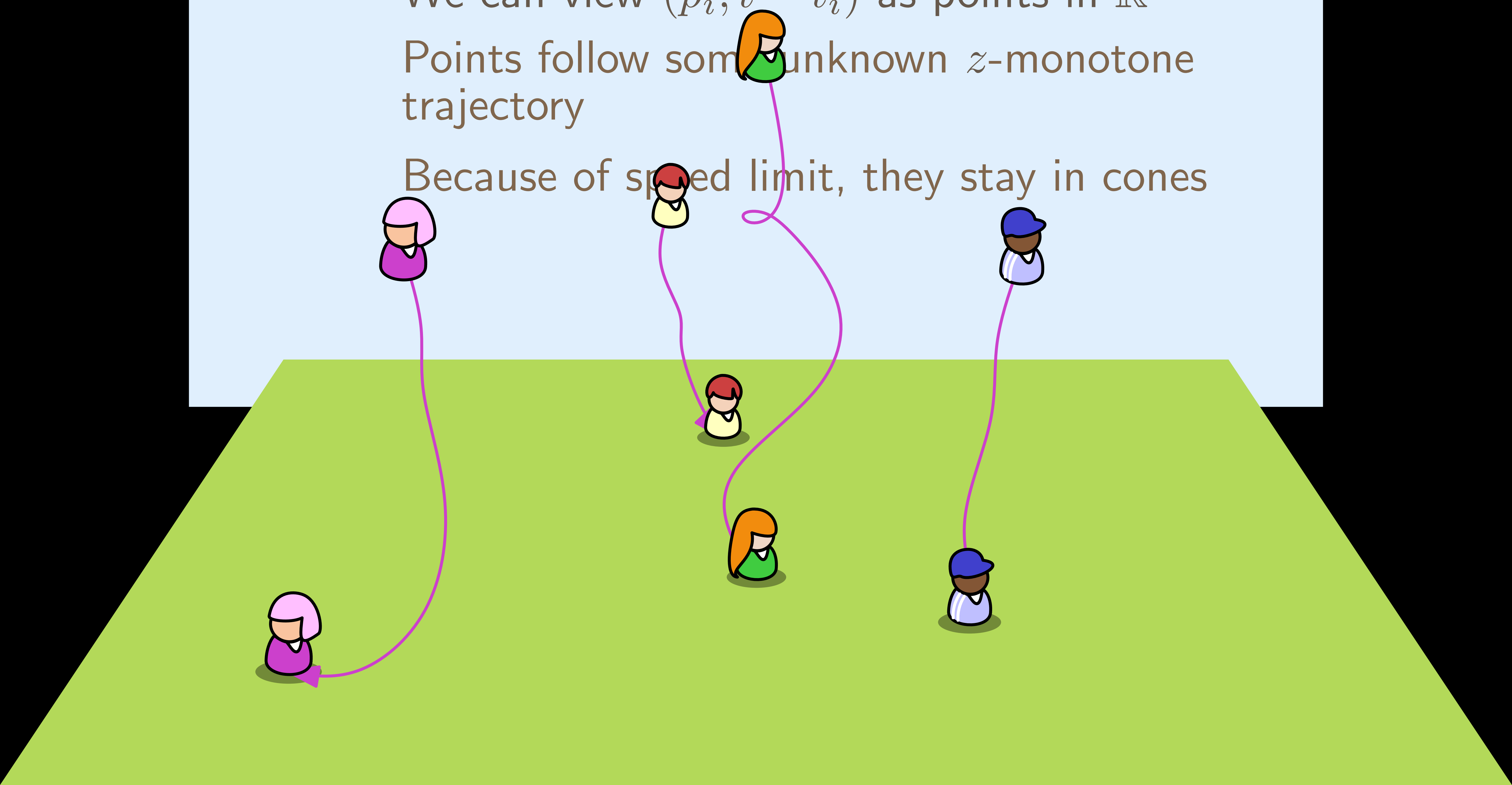
Consider n moving points

Suppose that for each point we know a *location* p_i and an associated *time stamp* t_i

We can view $(p_i, t - t_i)$ as points in $\mathbb{R}^3$

Points follow some unknown z-monotone trajectory

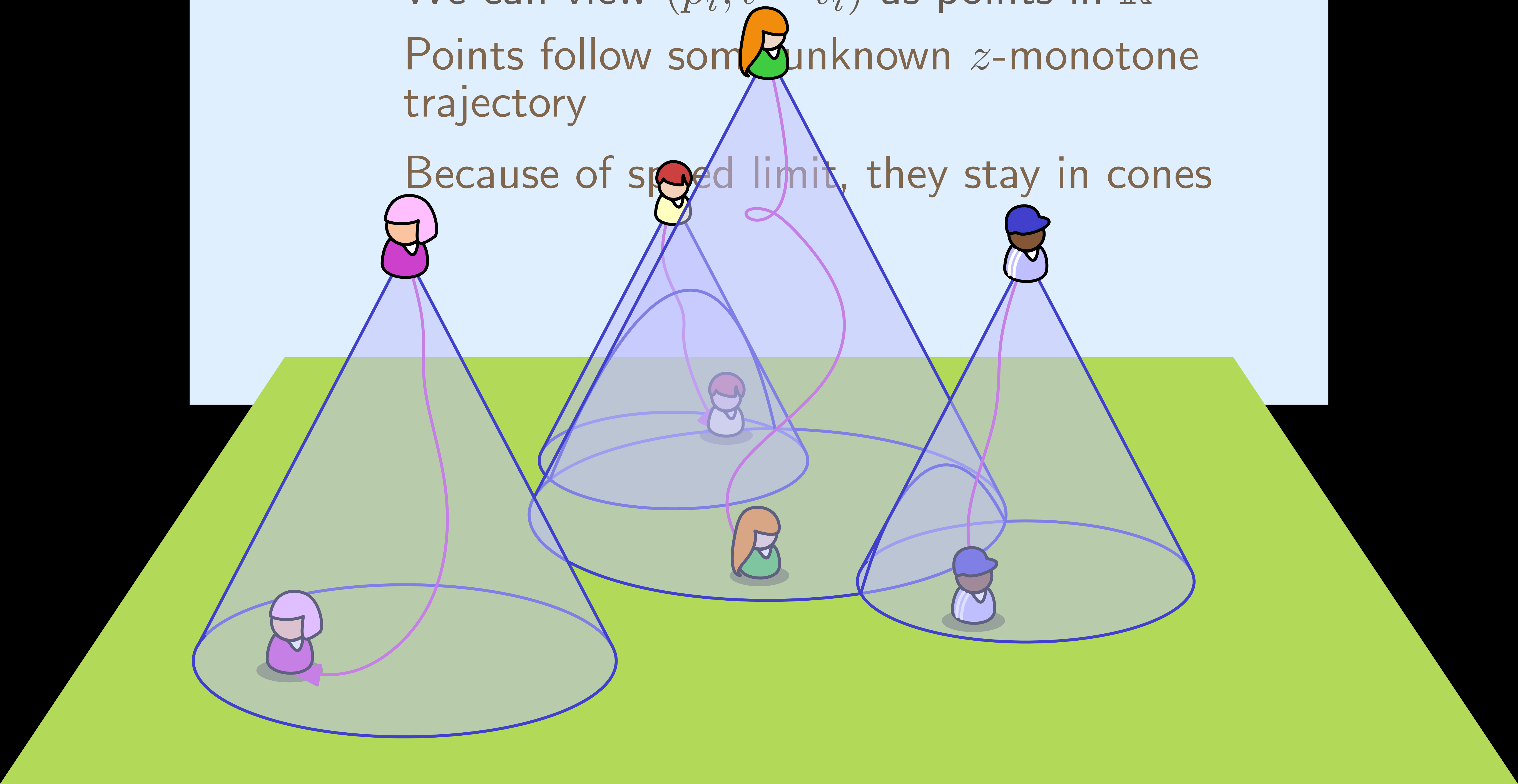
Because of speed limit, they stay in cones



UNCERTAIN POINTS

Consider n moving points

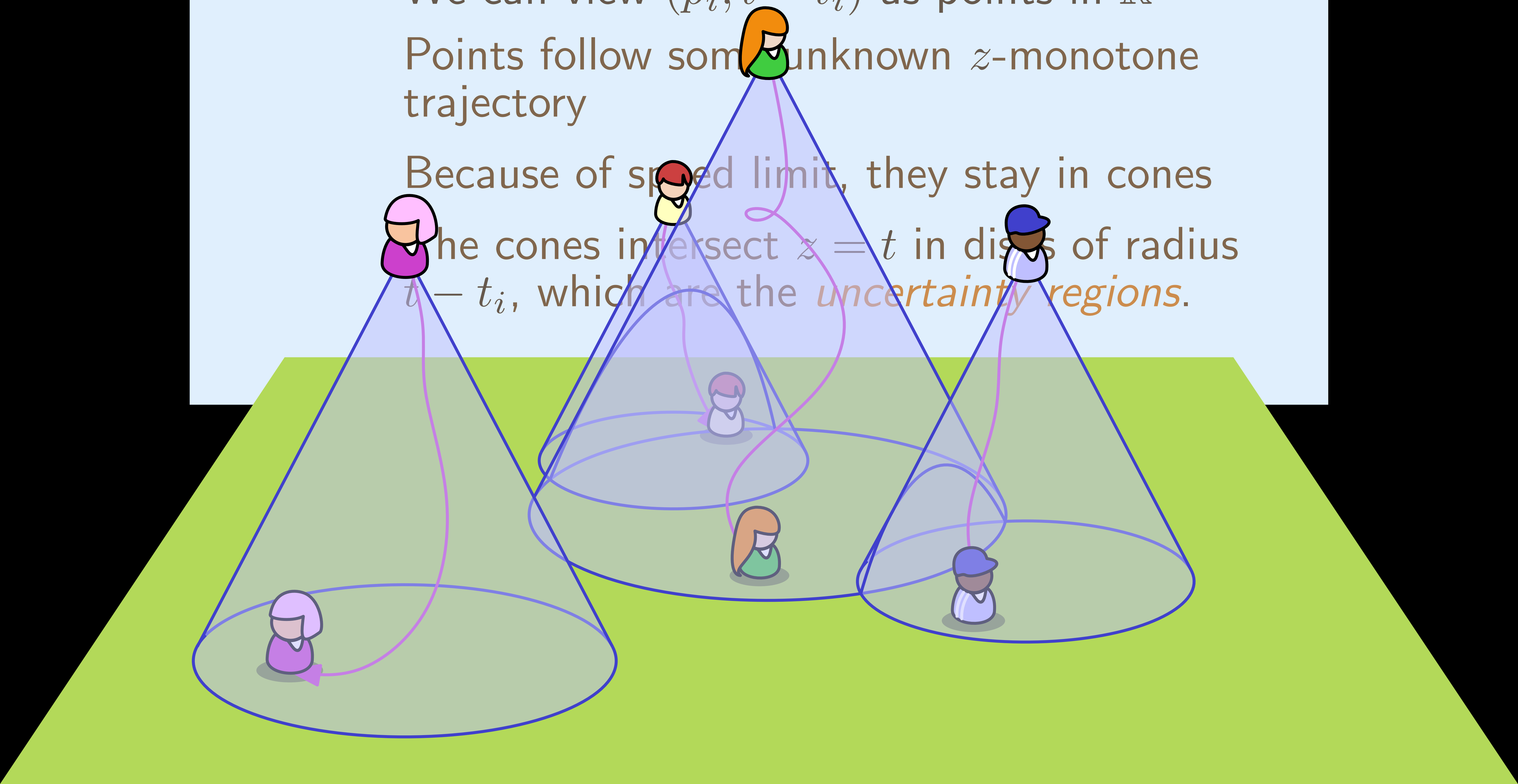
- Suppose that for each point we know a *location* p_i and an associated *time stamp* t_i
- We can view $(p_i, t - t_i)$ as points in $\mathbb{R}^3$
- Points follow some unknown z-monotone trajectory
- Because of speed limit, they stay in cones



UNCERTAIN POINTS

Consider n moving points

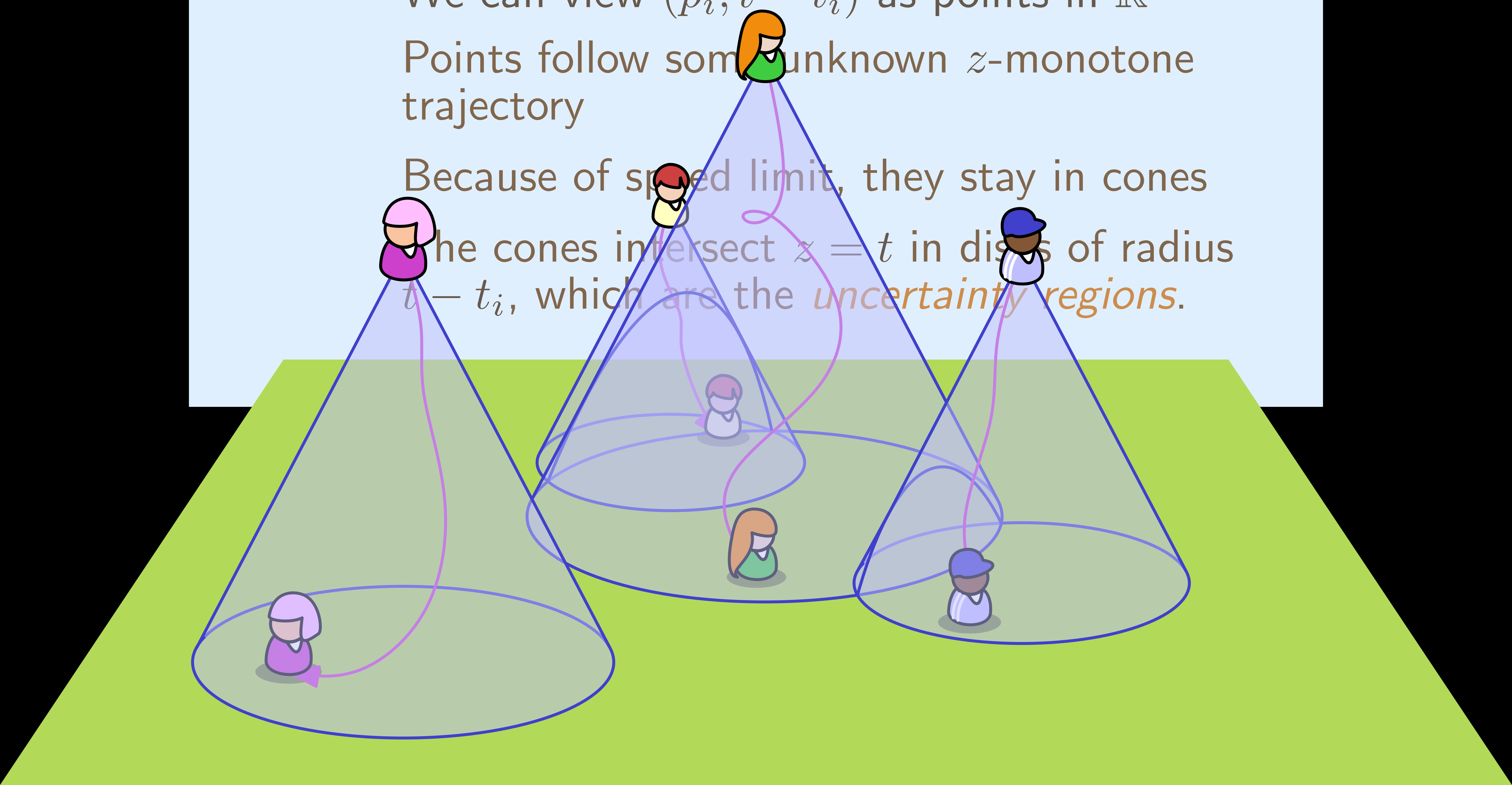
- Suppose that for each point we know a *location* p_i and an associated *time stamp* t_i
- We can view $(p_i, t - t_i)$ as points in $\mathbb{R}^3$
- Points follow some unknown z-monotone trajectory
- Because of speed limit, they stay in cones
- The cones intersect $z = t$ in discs of radius $t - t_i$, which are the *uncertainty regions*.



UNCERTAIN POINTS

Consider n moving points

- Suppose that for each point we know a *location* p_i and an associated *time stamp* t_i
- We can view $(p_i, t - t_i)$ as points in $\mathbb{R}^3$
- Points follow some unknown z-monotone trajectory
- Because of speed limit, they stay in cones
- The cones intersect $z = t$ in discs of radius $t - t_i$, which are the *uncertainty regions*.



UNCERTAIN POINTS

Consider n moving points

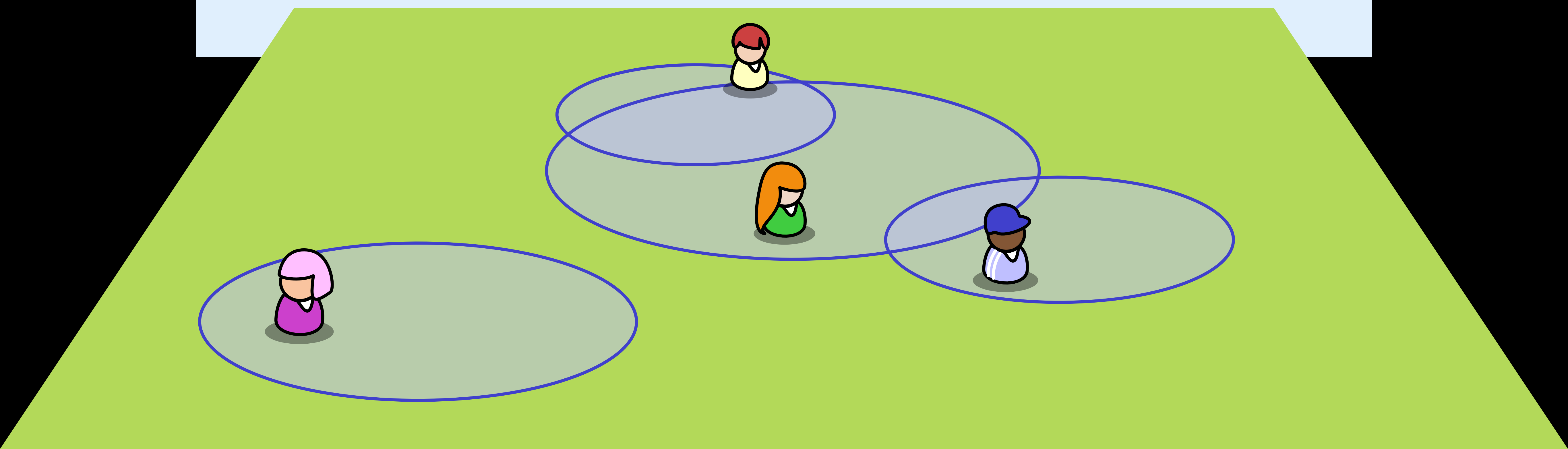
Suppose that for each point we know a *location* p_i and an associated *time stamp* t_i

We can view $(p_i, t - t_i)$ as points in $\mathbb{R}^3$

Points follow some unknown z-monotone trajectory

Because of speed limit, they stay in cones

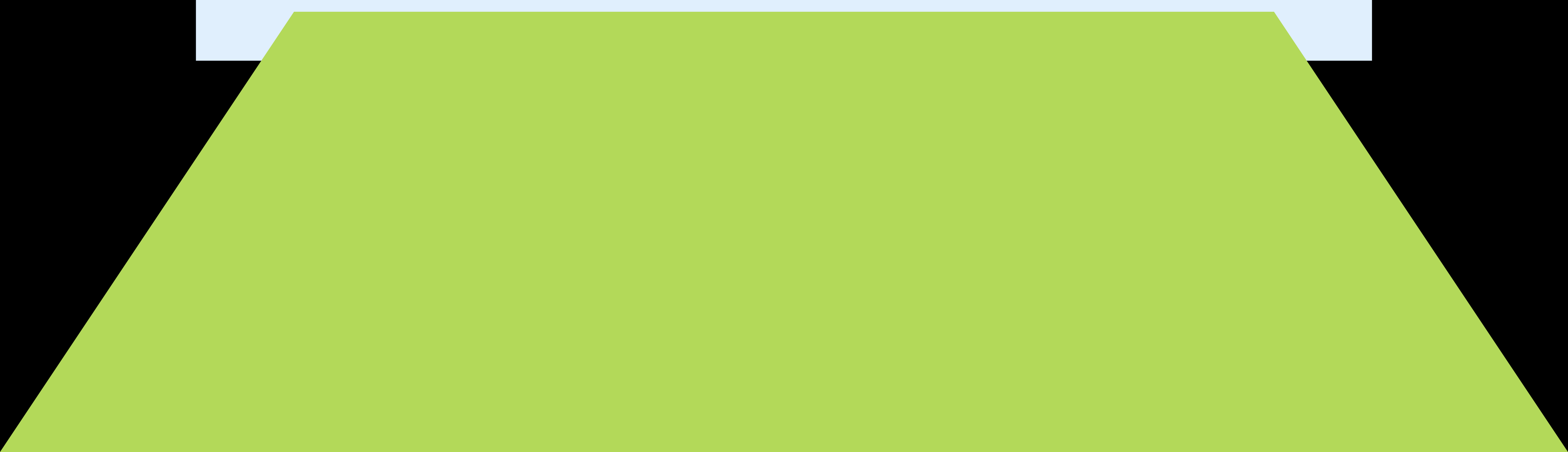
The cones intersect $z = t$ in disks of radius $t - t_i$, which are the *uncertainty regions*.



LOCATION QUERIES

LOCATION QUERIES

Consider n moving points



LOCATION QUERIES

Consider n moving points

LOCATION QUERIES

Consider n moving points

Suppose that for each point we know a *location* p_i and an associated *time stamp* t_i

LOCATION QUERIES

Consider n moving points

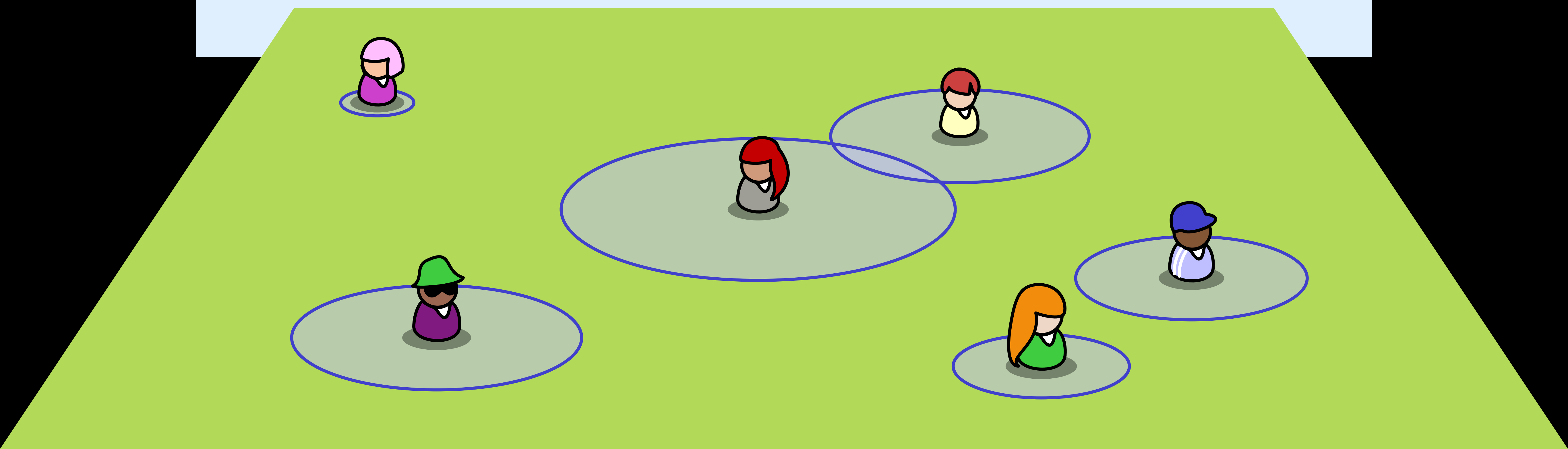
Suppose that for each point we know a *location* p_i and an associated *time stamp* t_i

Then the current uncertainty region of each point is a disk centered at p_i

LOCATION QUERIES

Consider n moving points

- Suppose that for each point we know a *location* p_i and an associated *time stamp* t_i
- Then the current uncertainty region of each point is a disk centered at p_i



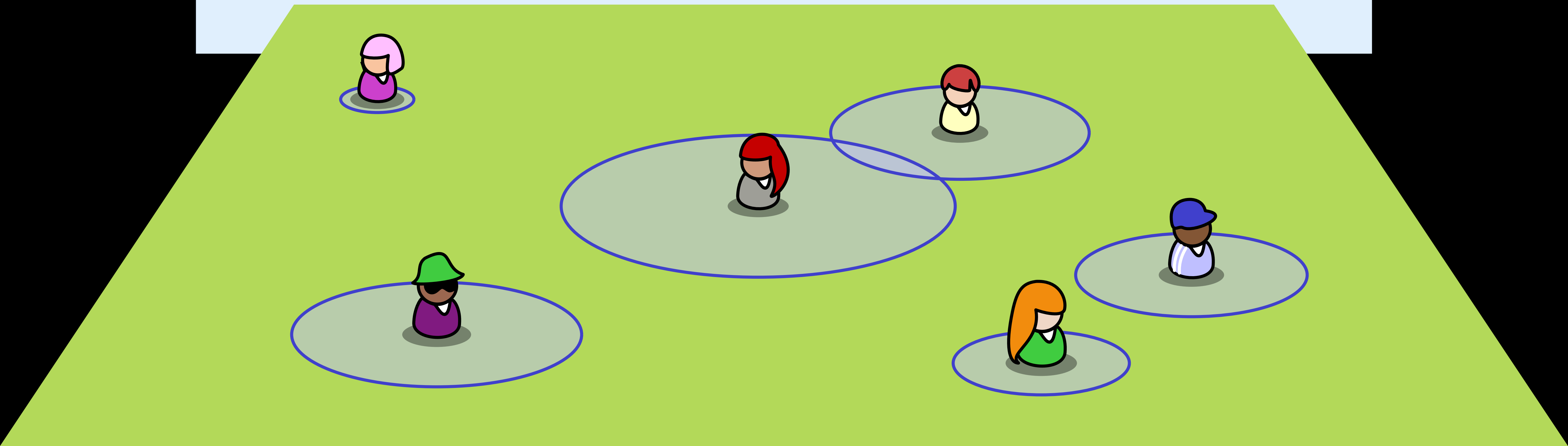
LOCATION QUERIES

Consider n moving points

Suppose that for each point we know a *location* p_i and an associated *time stamp* t_i

Then the current uncertainty region of each point is a disk centered at p_i

Each unit of time, all regions grow by d



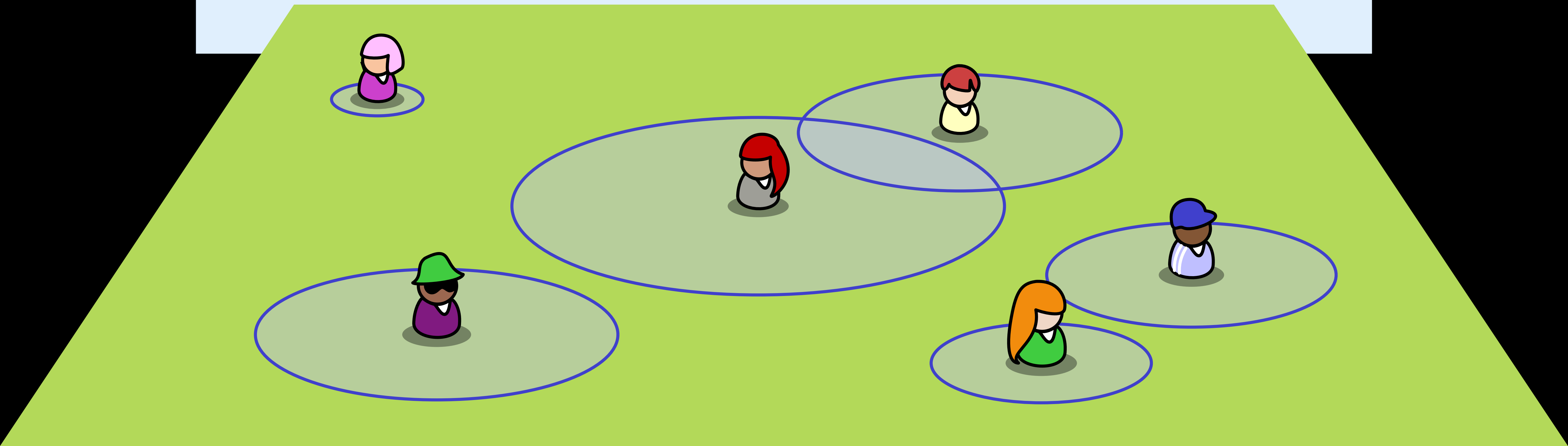
LOCATION QUERIES

Consider n moving points

Suppose that for each point we know a *location* p_i and an associated *time stamp* t_i

Then the current uncertainty region of each point is a disk centered at p_i

Each unit of time, all regions grow by d



LOCATION QUERIES

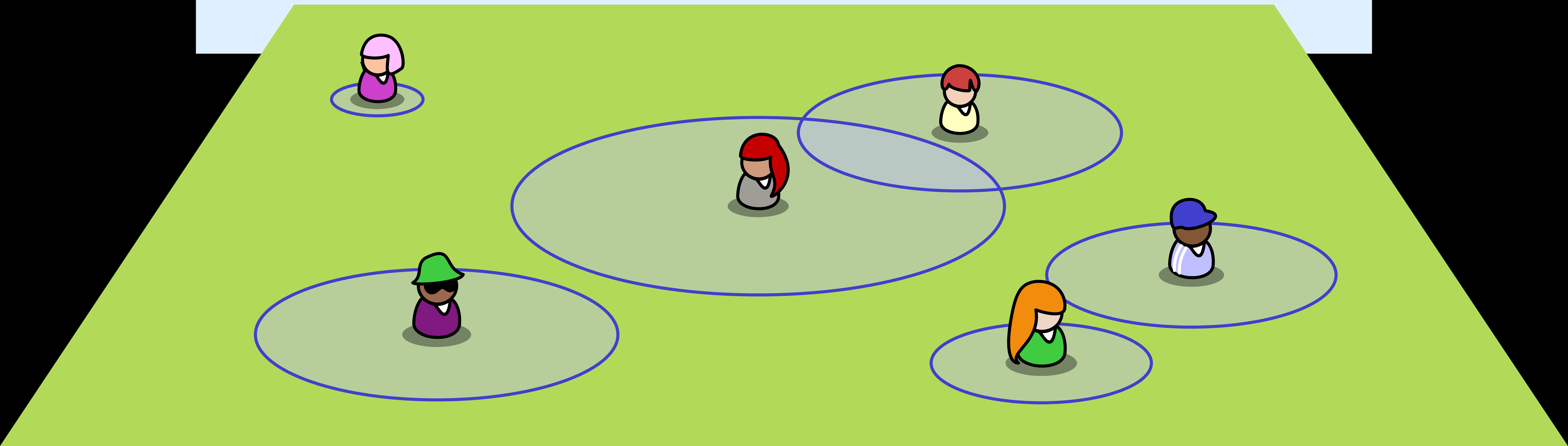
Consider n moving points

Suppose that for each point we know a *location* p_i and an associated *time stamp* t_i

Then the current uncertainty region of each point is a disk centered at p_i

Each unit of time, all regions grow by d

We can *query* a point to update it



LOCATION QUERIES

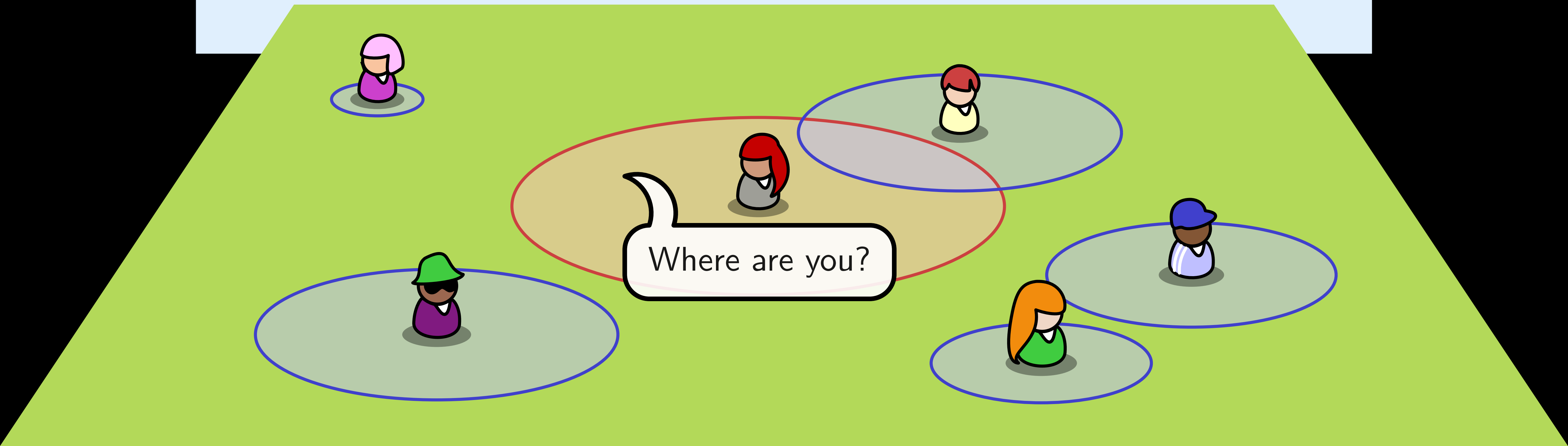
Consider n moving points

Suppose that for each point we know a *location* p_i and an associated *time stamp* t_i

Then the current uncertainty region of each point is a disk centered at p_i

Each unit of time, all regions grow by d

We can *query* a point to update it



LOCATION QUERIES

- Consider n moving points
  - Suppose that for each point we know a *location* p_i and an associated *time stamp* t_i
  - Then the current uncertainty region of each point is a disk centered at p_i
  - Each unit of time, all regions grow by d
- We can *query* a point to update it
  - Takes 1 unit of time to execute

LOCATION QUERIES

Consider n moving points

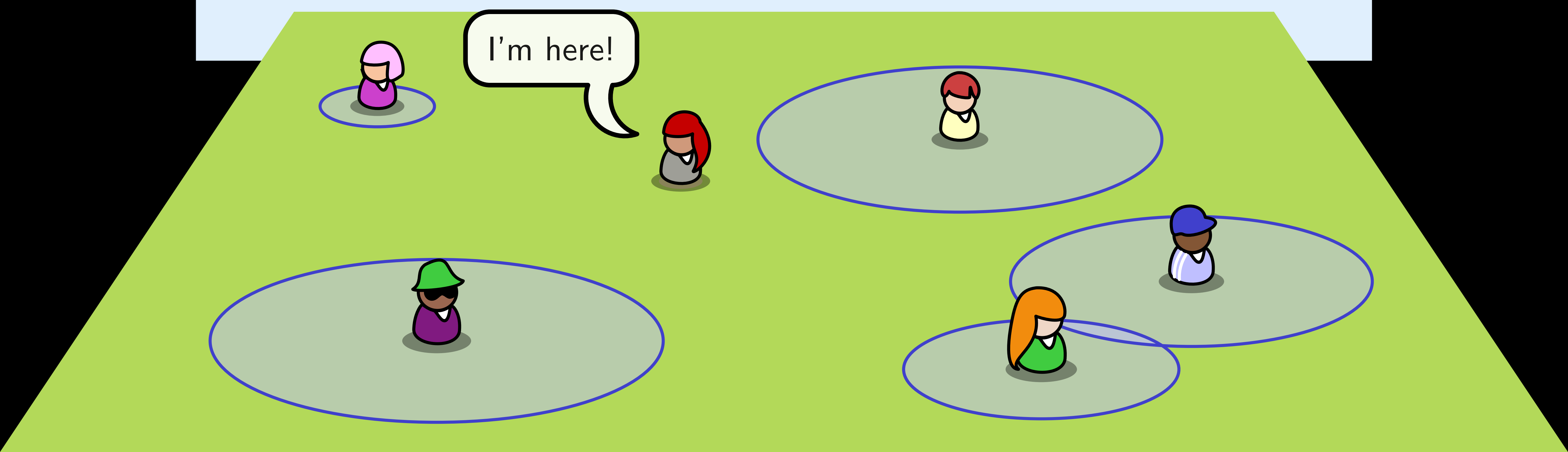
Suppose that for each point we know a *location* p_i and an associated *time stamp* t_i

Then the current uncertainty region of each point is a disk centered at p_i

Each unit of time, all regions grow by d

We can *query* a point to update it

Takes 1 unit of time to execute



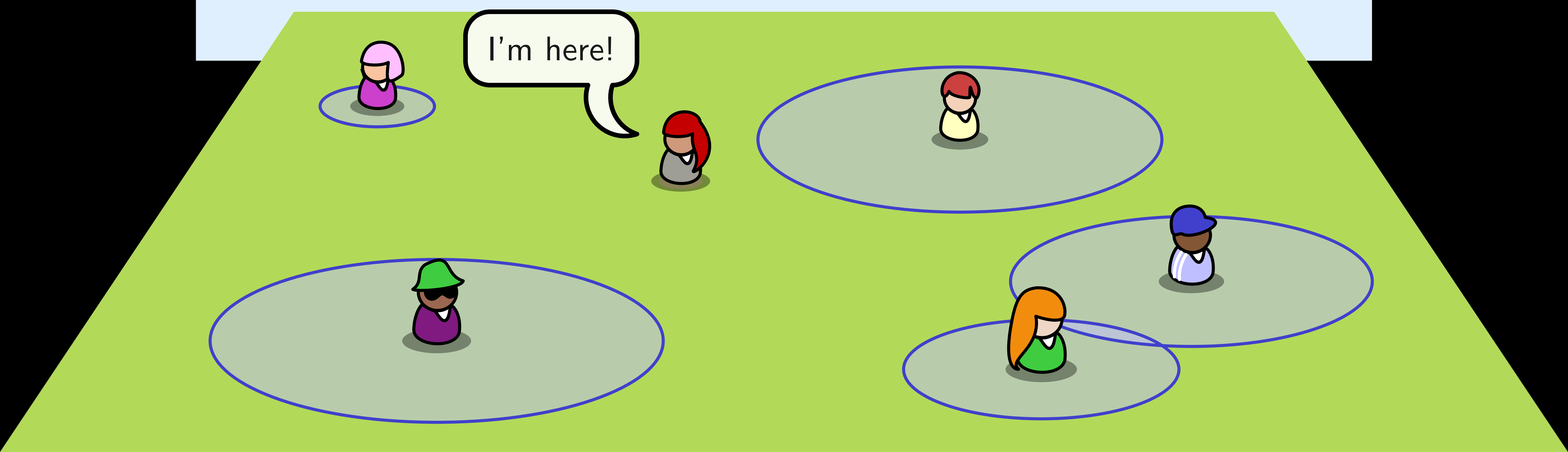
LOCATION QUERIES

Consider n moving points

- Suppose that for each point we know a *location* p_i and an associated *time stamp* t_i
- Then the current uncertainty region of each point is a disk centered at p_i
- Each unit of time, all regions grow by d

We can *query* a point to update it

- Takes 1 unit of time to execute
- After a query, one region collapses to a point, but all other regions grow



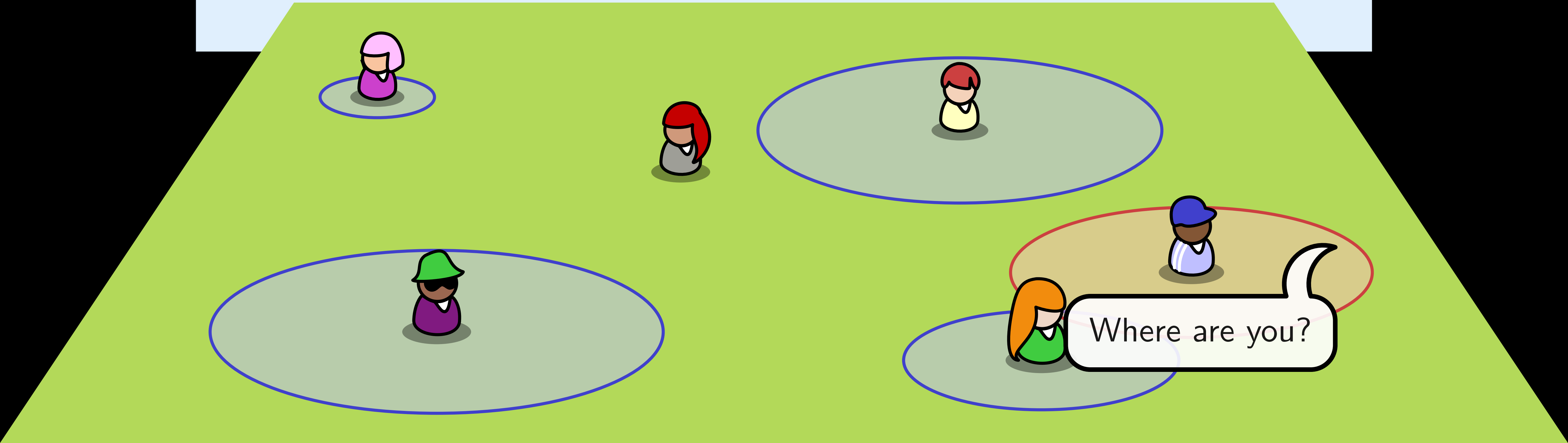
LOCATION QUERIES

Consider n moving points

- Suppose that for each point we know a *location* p_i and an associated *time stamp* t_i
- Then the current uncertainty region of each point is a disk centered at p_i
- Each unit of time, all regions grow by d

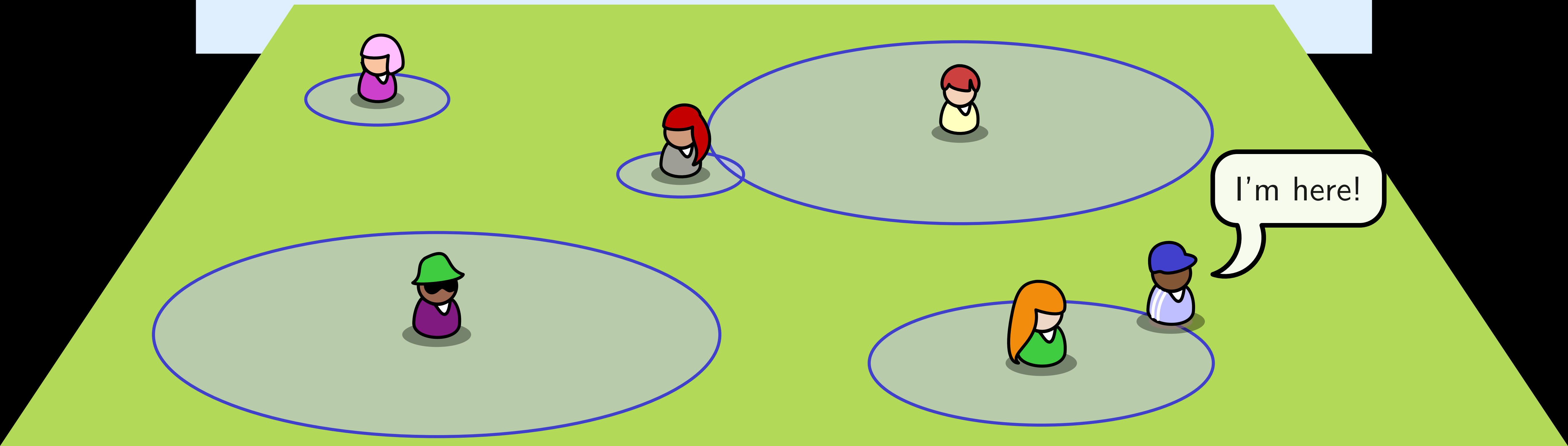
We can *query* a point to update it

- Takes 1 unit of time to execute
- After a query, one region collapses to a point, but all other regions grow



LOCATION QUERIES

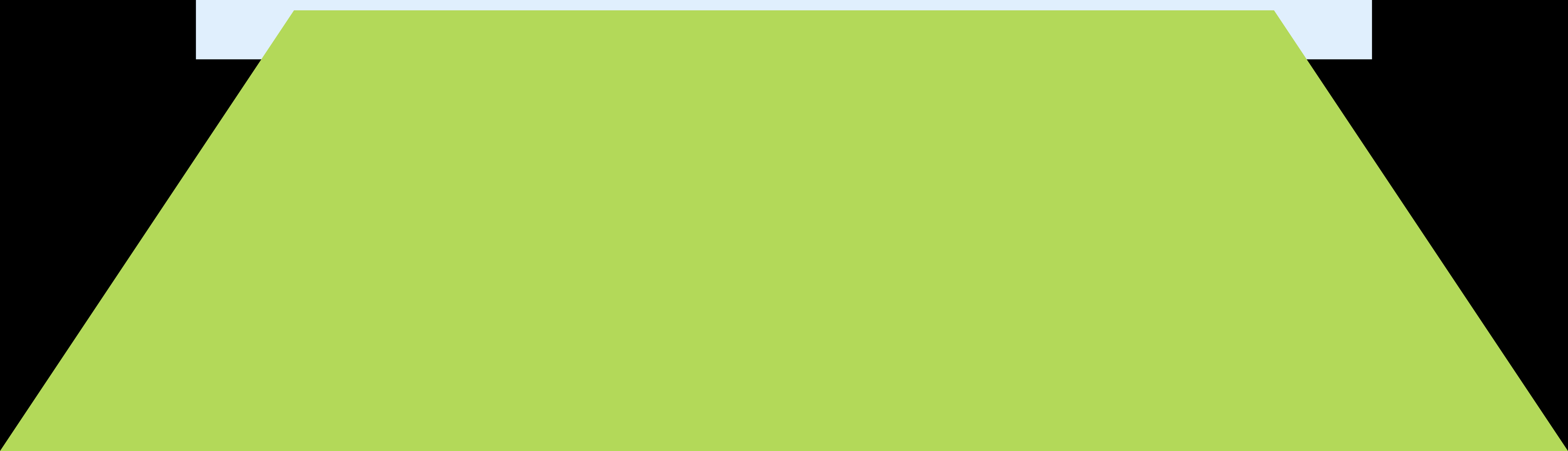
- Consider n moving points
  - Suppose that for each point we know a *location* p_i and an associated *time stamp* t_i
  - Then the current uncertainty region of each point is a disk centered at p_i
  - Each unit of time, all regions grow by d
- We can *query* a point to update it
  - Takes 1 unit of time to execute
  - After a query, one region collapses to a point, but all other regions grow



CONGESTION MEASURES

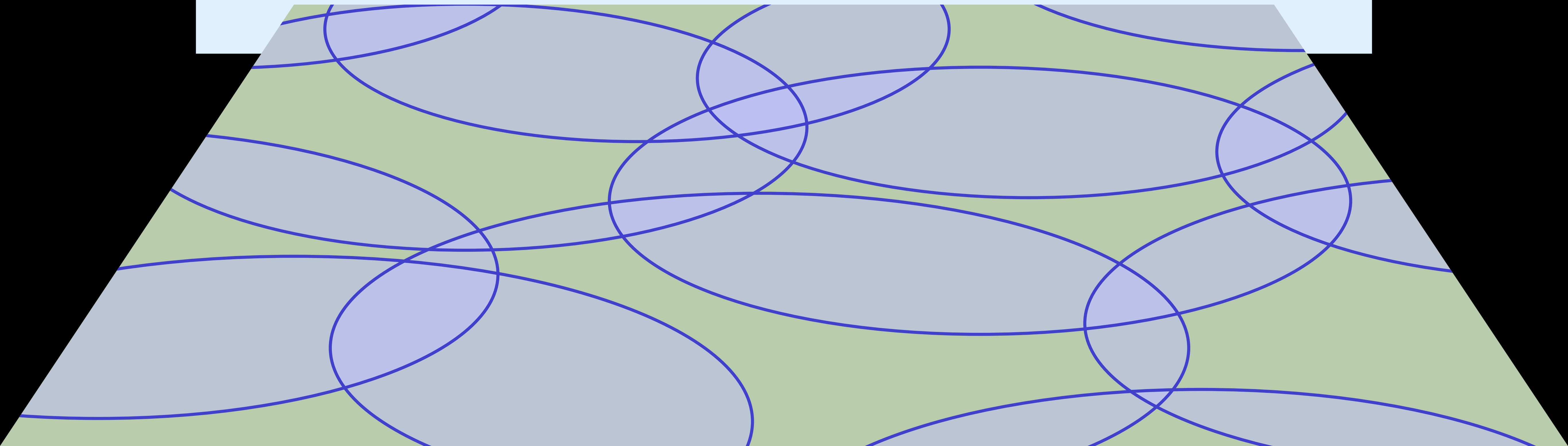
CONGESTION MEASURES

Consider a set of regions



CONGESTION MEASURES

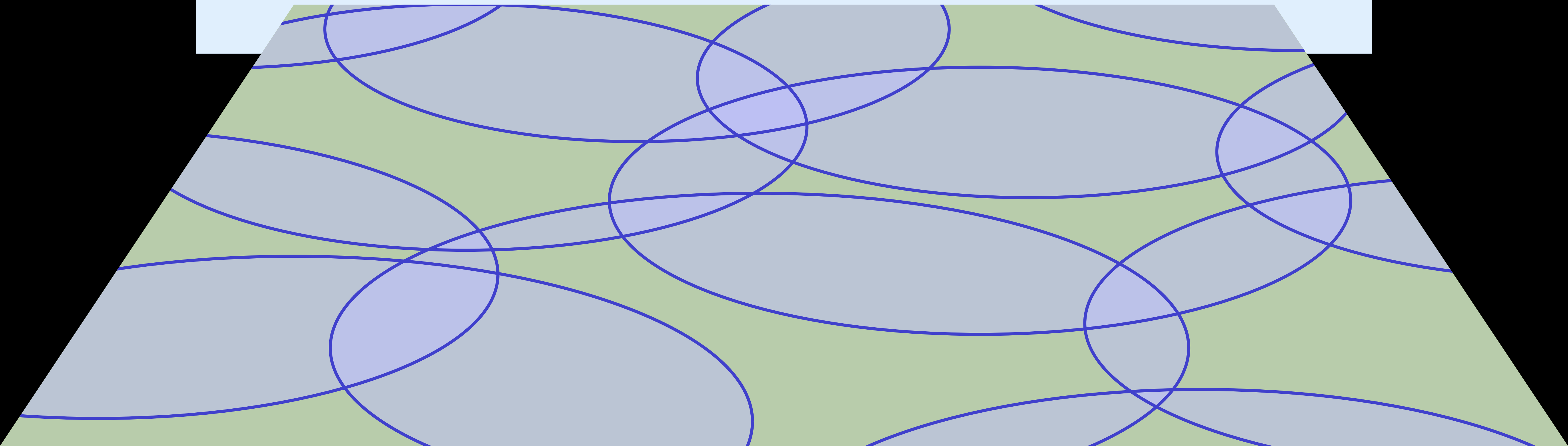
Consider a set of regions



CONGESTION MEASURES

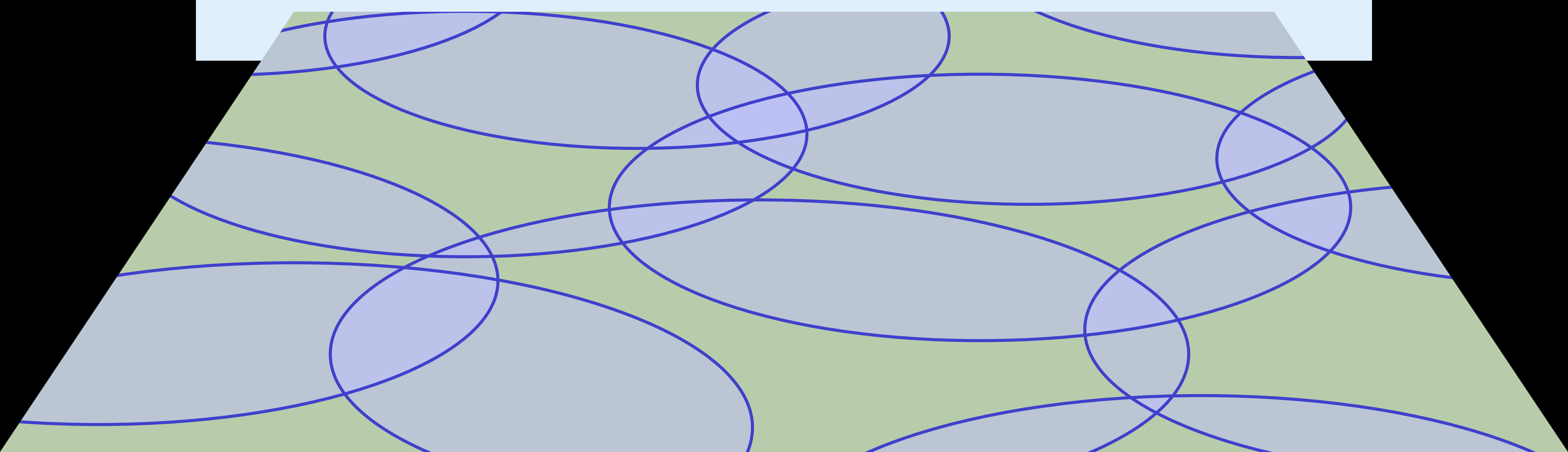
Consider a set of regions

What do they tell us about the point set?



CONGESTION MEASURES

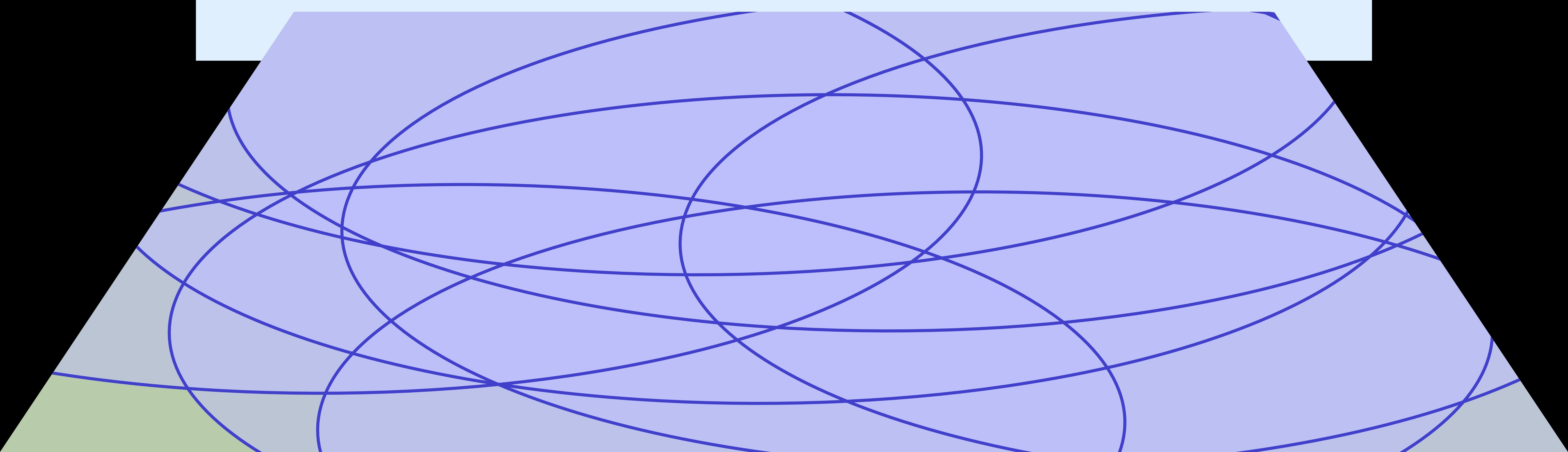
- Consider a set of regions
  - What do they tell us about the point set?
  - The more they overlap, the less information we have.



CONGESTION MEASURES

Consider a set of regions

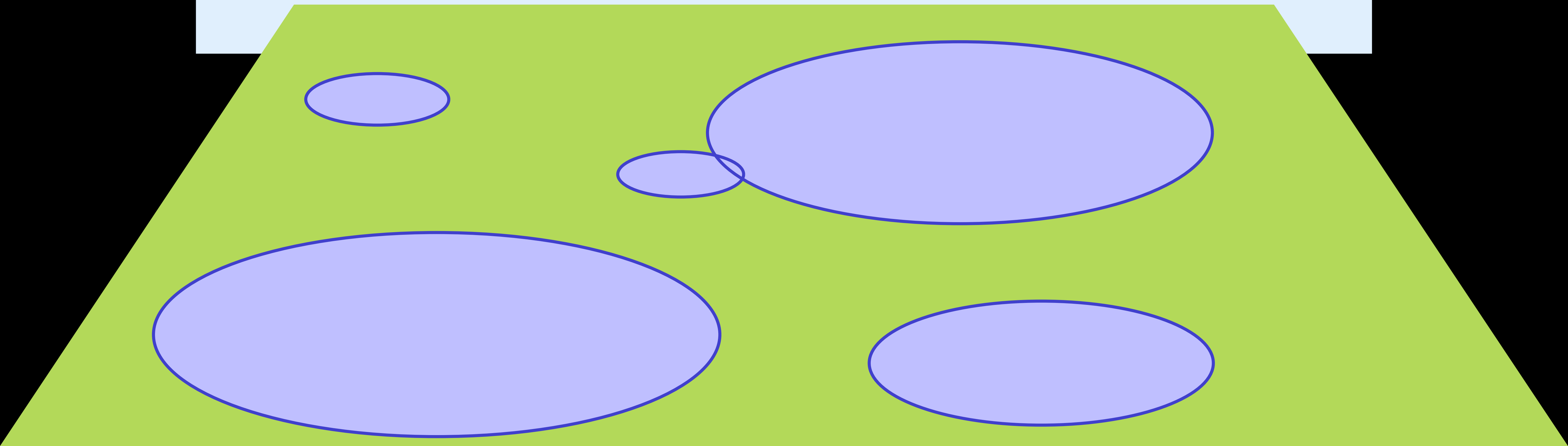
- What do they tell us about the point set?
- The more they overlap, the less information we have.



CONGESTION MEASURES

Consider a set of regions

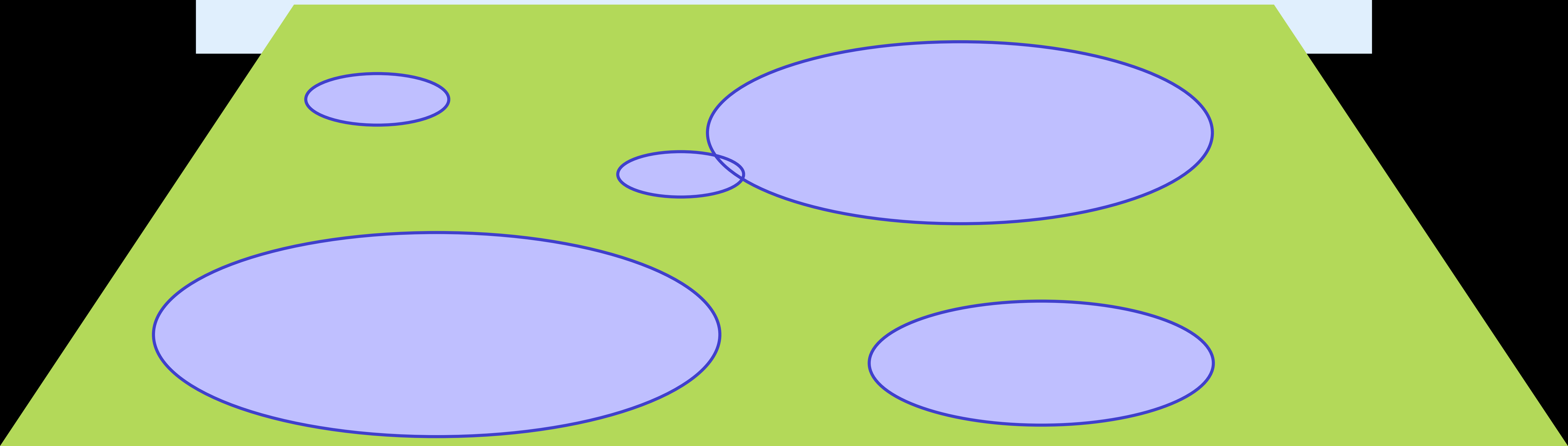
- What do they tell us about the point set?
- The more they overlap, the less information we have.



CONGESTION MEASURES

Consider a set of regions

- What do they tell us about the point set?
- The more they overlap, the less information we have.
- How do we measure how much a set of regions is "overlapping"?



PLY

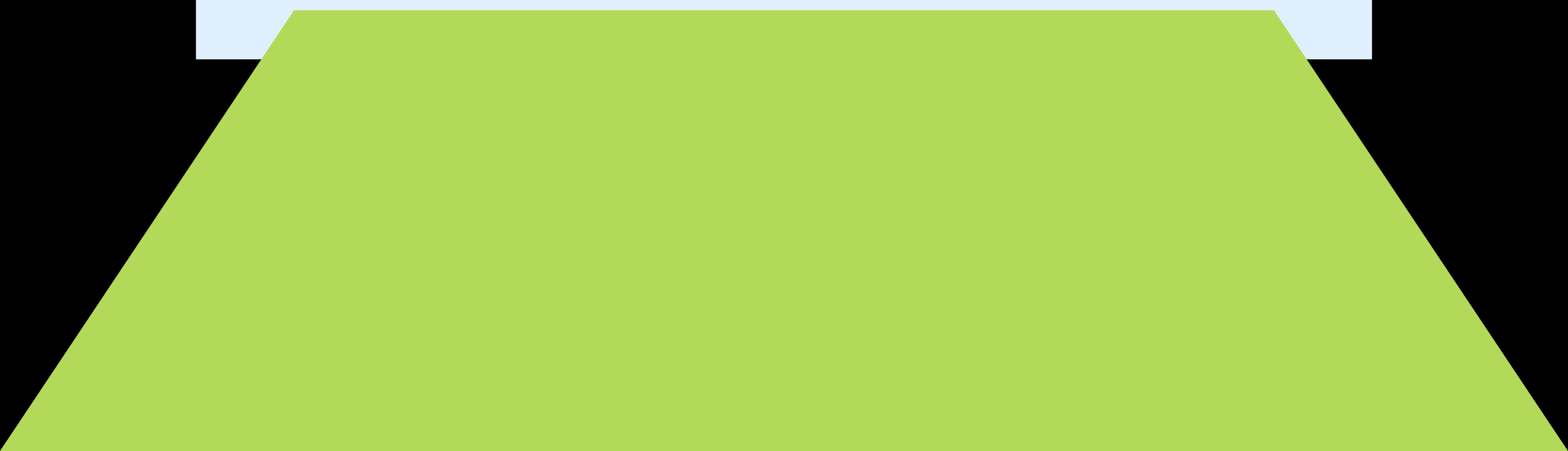
PLY

Ply of a point

PLY

Ply of a point

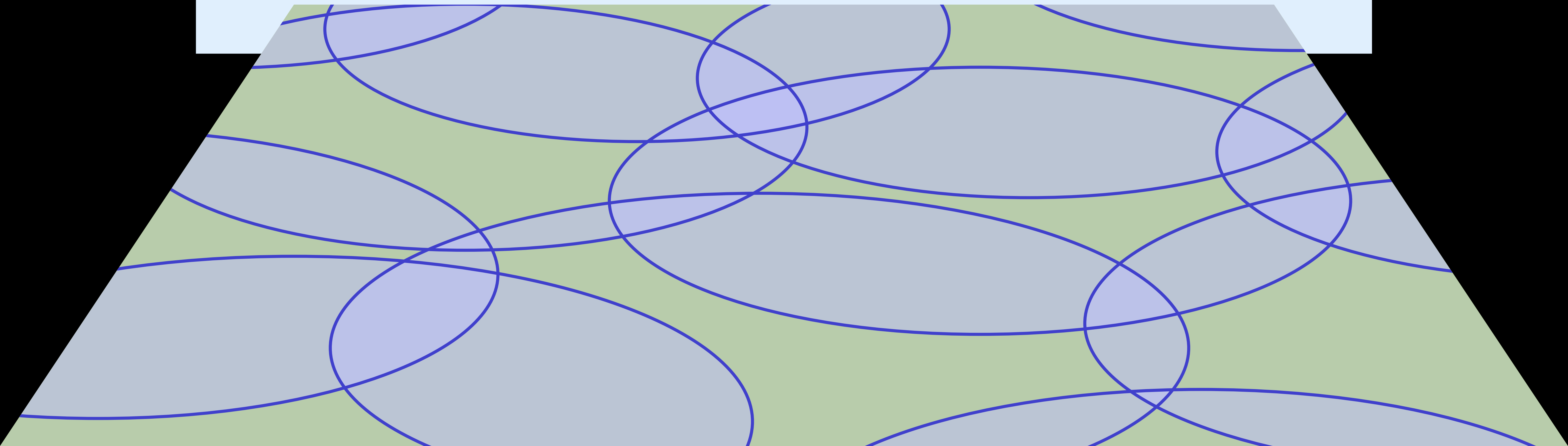
Given a set of regions



PLY

Ply of a point

Given a set of regions

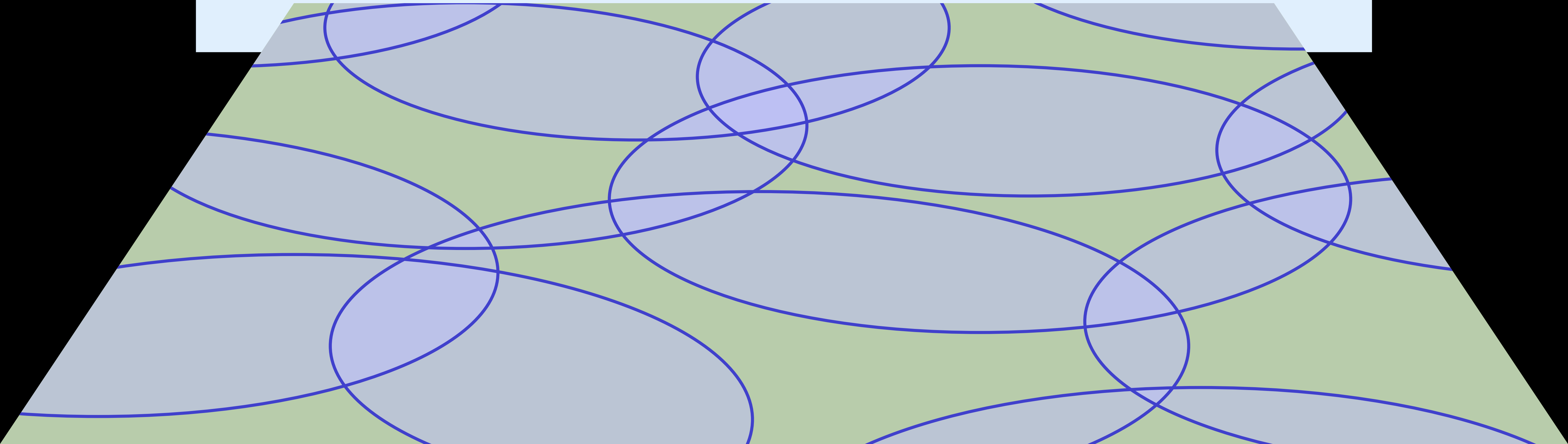


PLY

Ply of a point

Given a set of regions

$\delta(p) = |\{R : p \in R\}|$

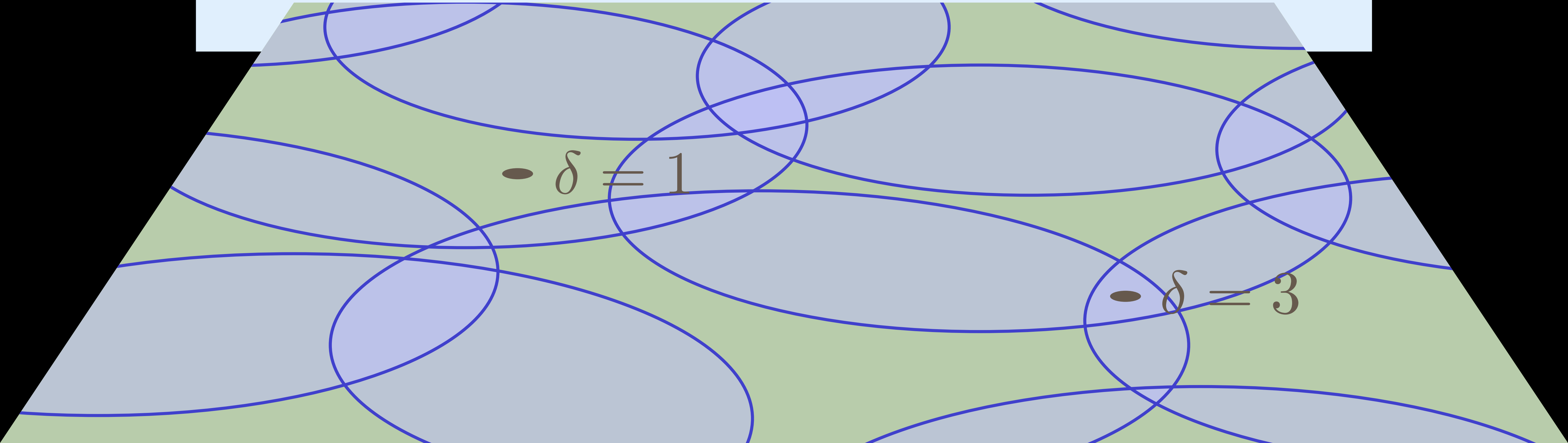


PLY

Ply of a point

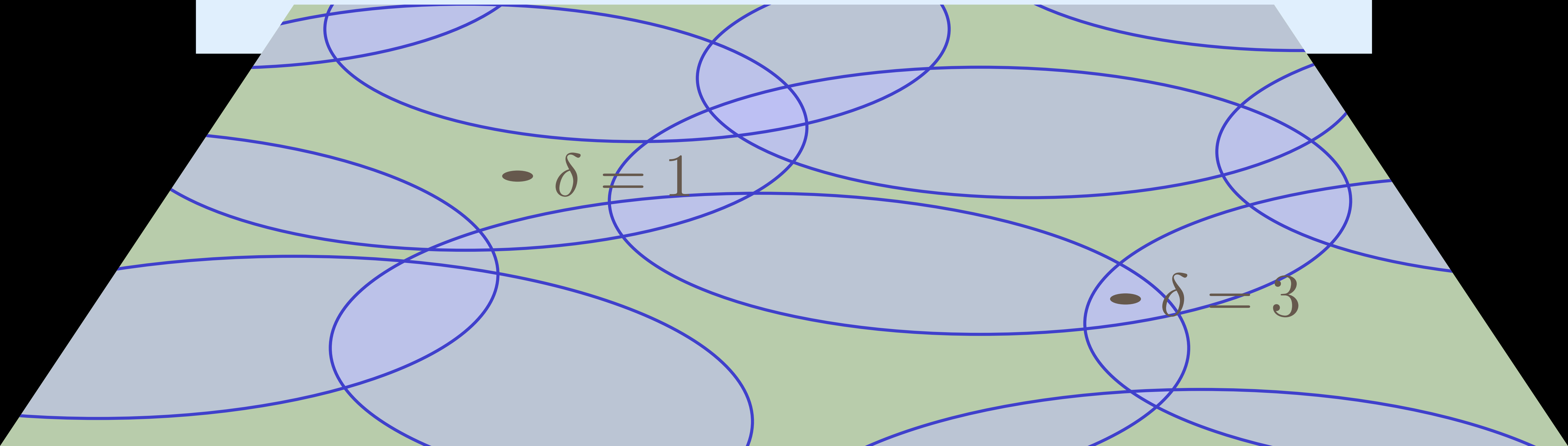
Given a set of regions

$\delta(p) = |\{R : p \in R\}|$



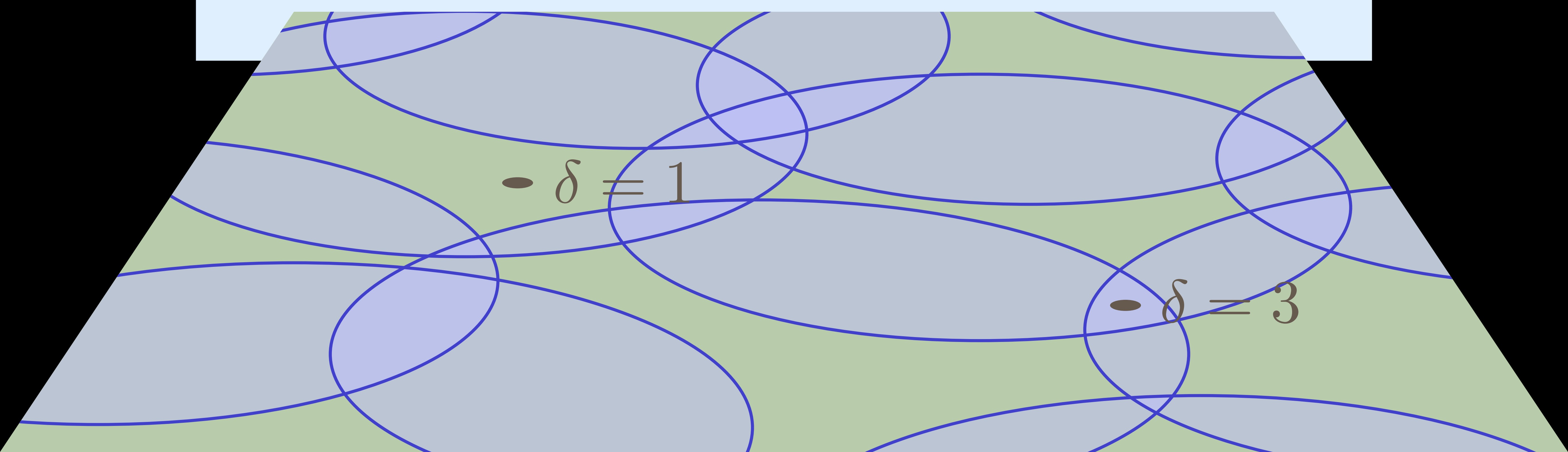
PLY

- Ply of a point
  - Given a set of regions
  - $\delta(p) = |\{R : p \in R\}|$
- Ply of the set of regions



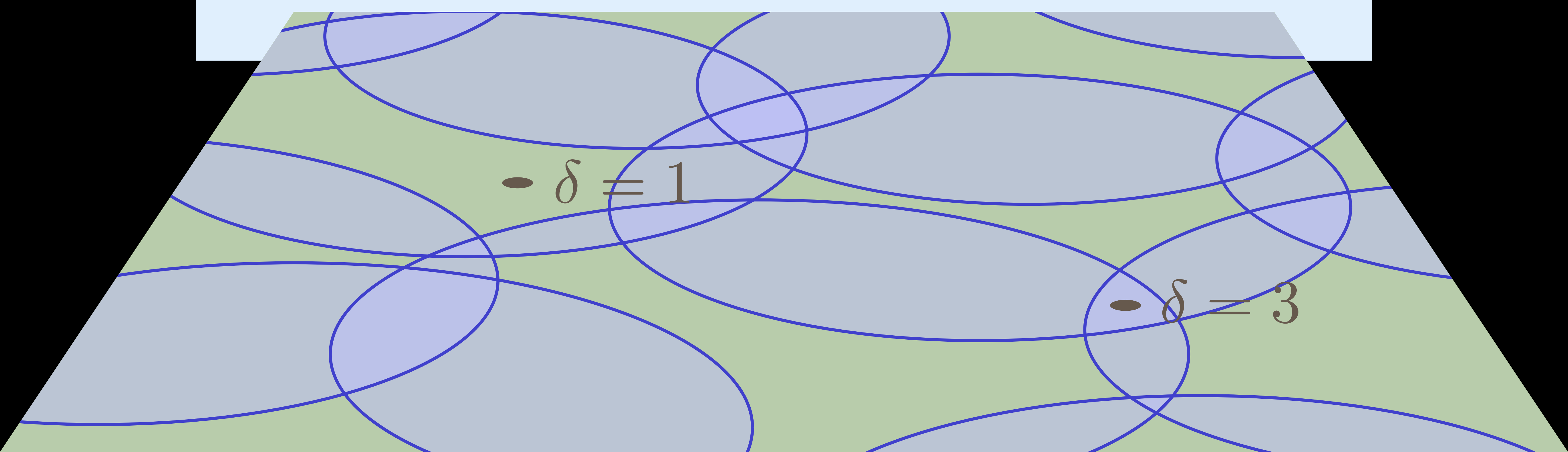
PLY

- 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



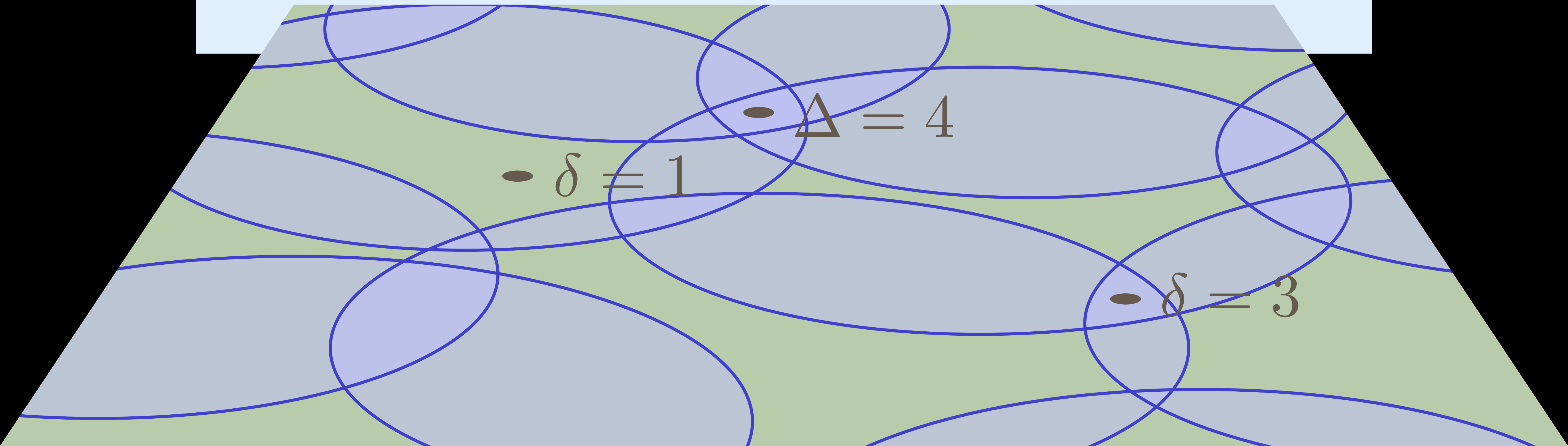
PLY

- 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

- 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

Ply of a point

Given a set of regions

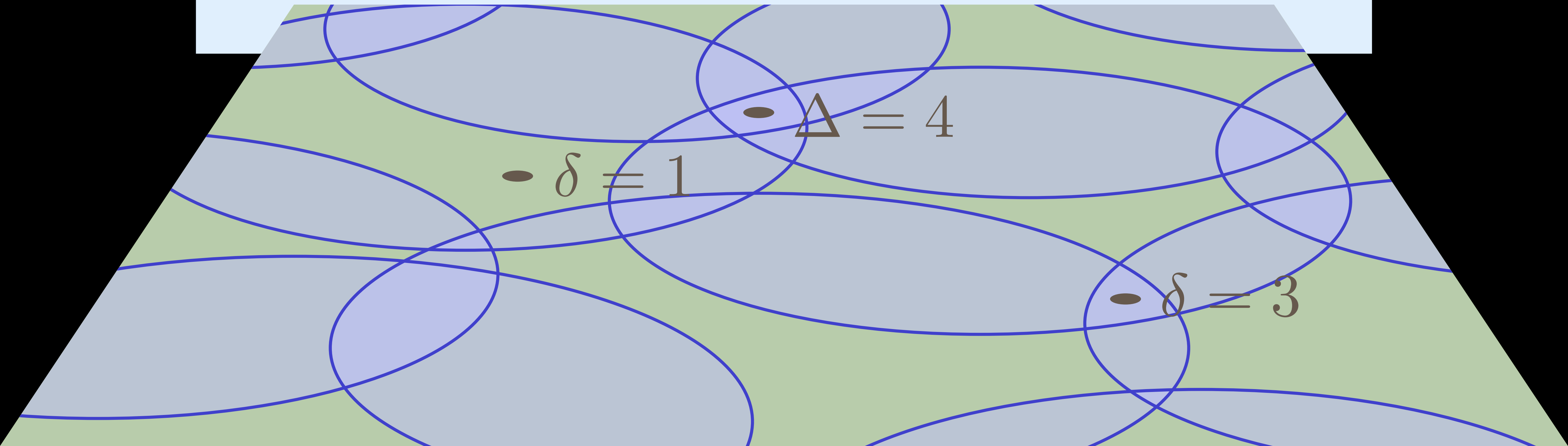
$\delta(p) = |\{R : p \in R\}|$

Ply of the set of regions

Maximum ply of all points

$\Delta = \max_p \delta(p)$

Low ply is good!



PLY

Ply of a point

Given a set of regions

$\delta(p) = |\{R : p \in R\}|$

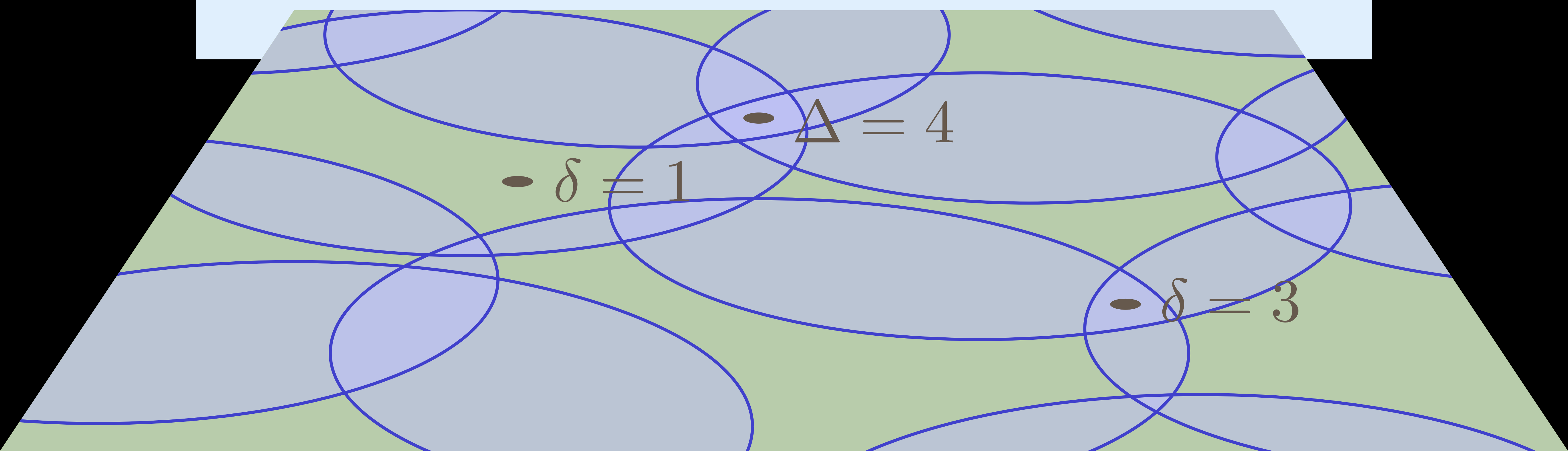
Ply of the set of regions

Maximum ply of all points

$\Delta = \max_p \delta(p)$

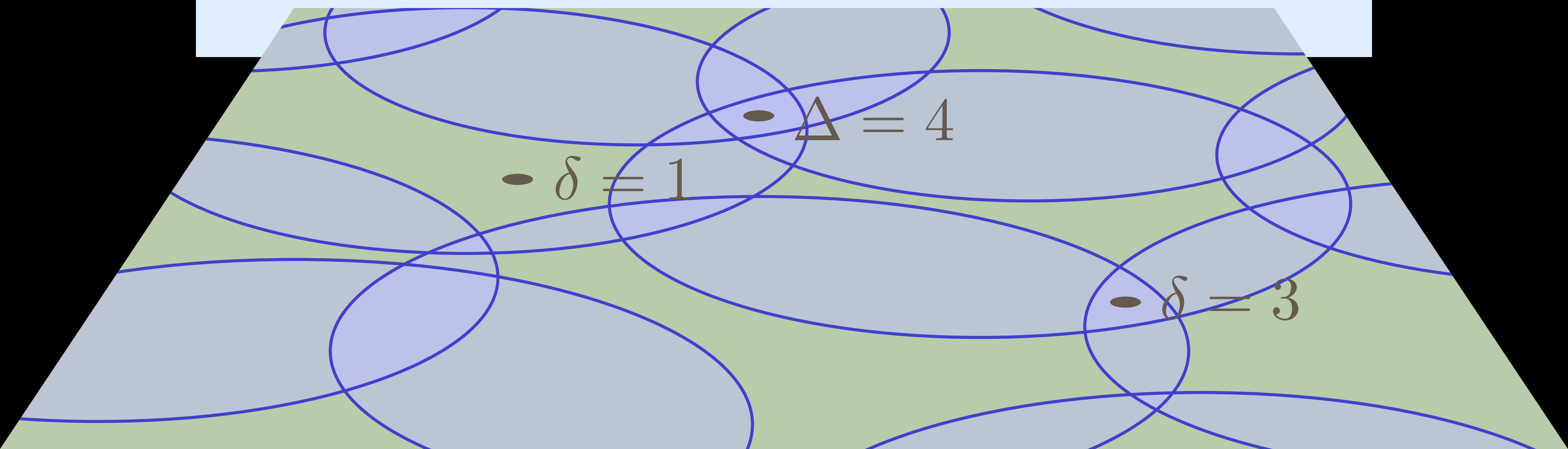
Low ply is good!

High ply is not *necessarily* bad.

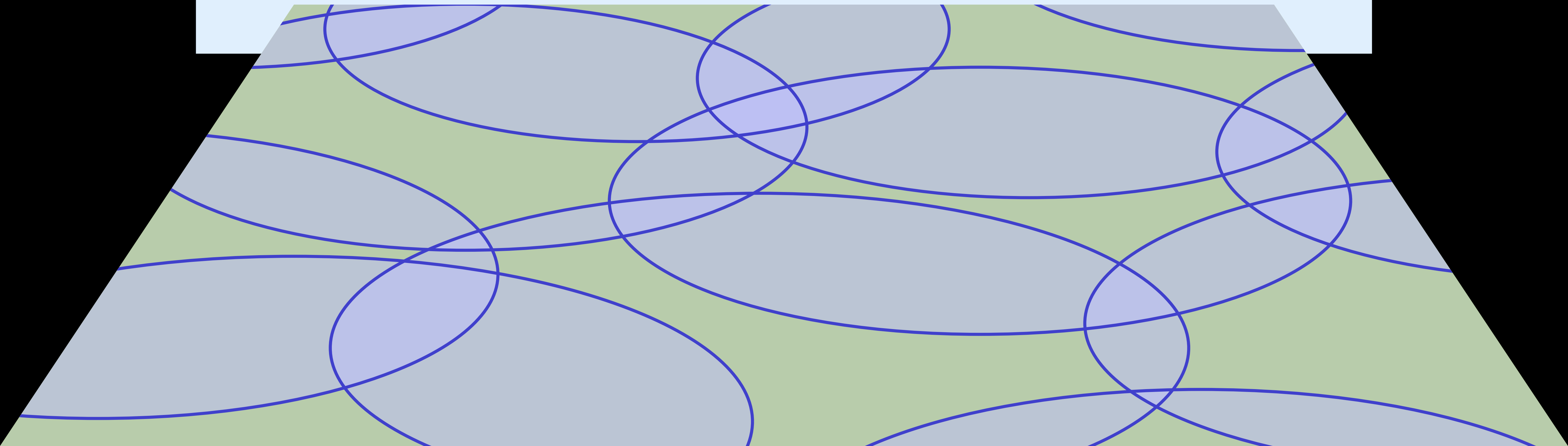


PLY

- 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
  - $\Delta = \max_p \delta(p)$
- Low ply is good!
- High ply is not *necessarily* bad.
- Ply is a *local* congestion measure.

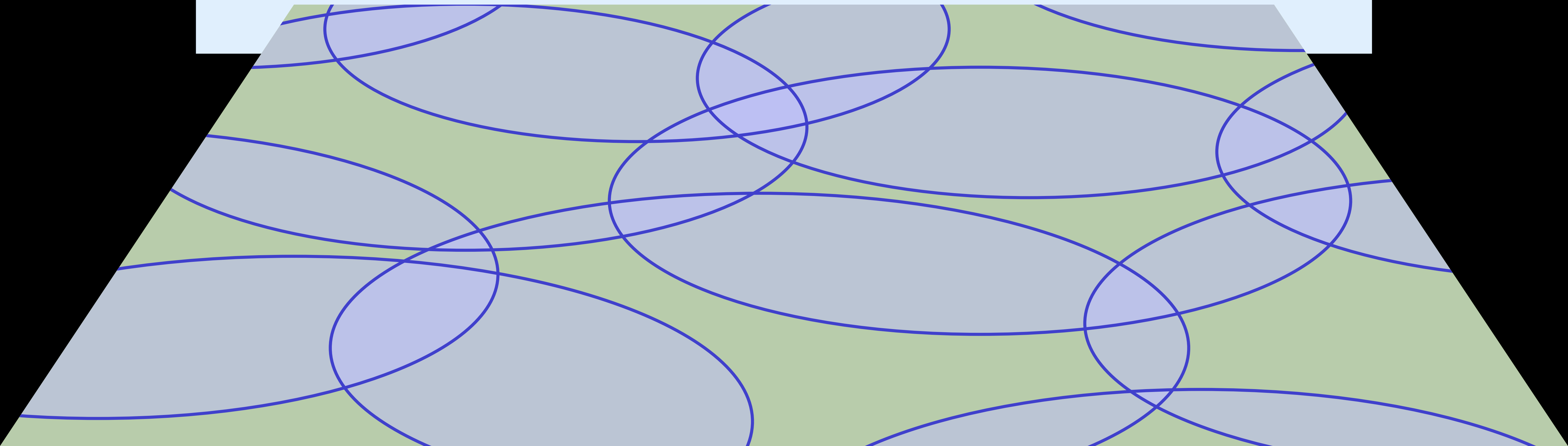


TOTAL DEGREE



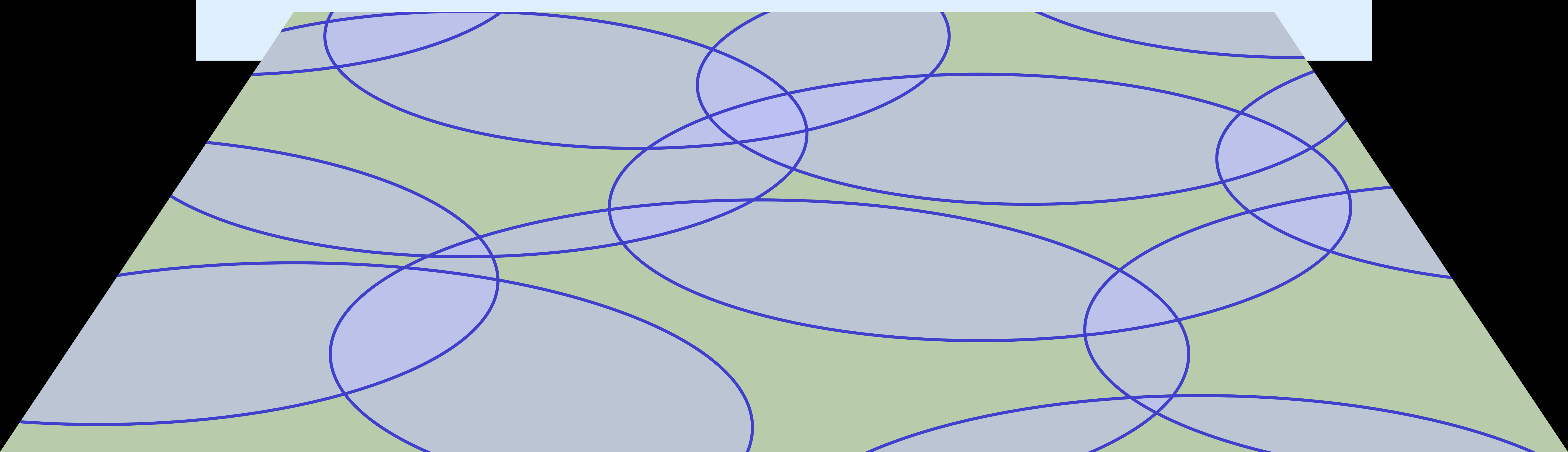
TOTAL DEGREE

Degree of a region



TOTAL DEGREE

- Degree of a region
  - Number of other regions is intersects

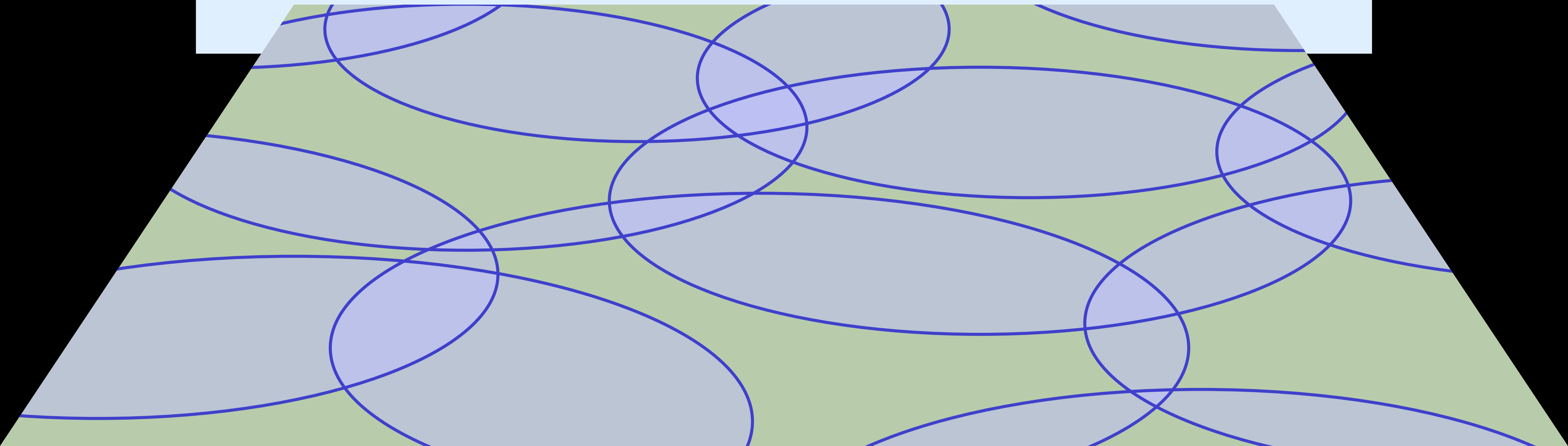


TOTAL DEGREE

Degree of a region

Number of other regions is intersects

$d(R) = |\{R' : R \cap R \neq \emptyset\}|$

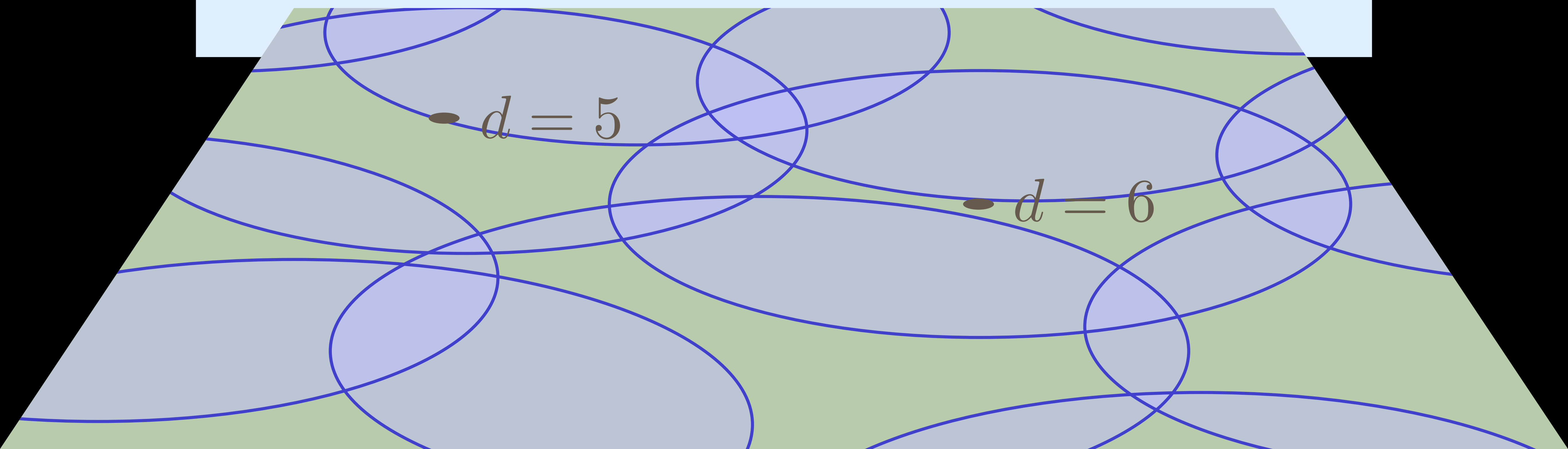


TOTAL DEGREE

Degree of a region

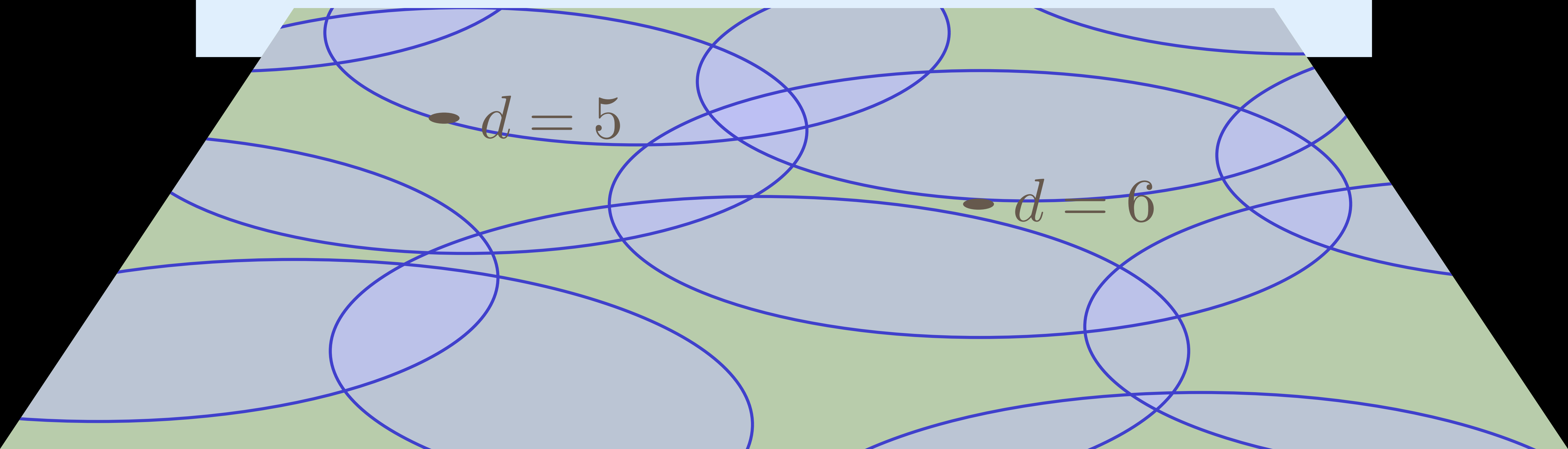
Number of other regions is intersects

$d(R) = |\{R' : R \cap R \neq \emptyset\}|$



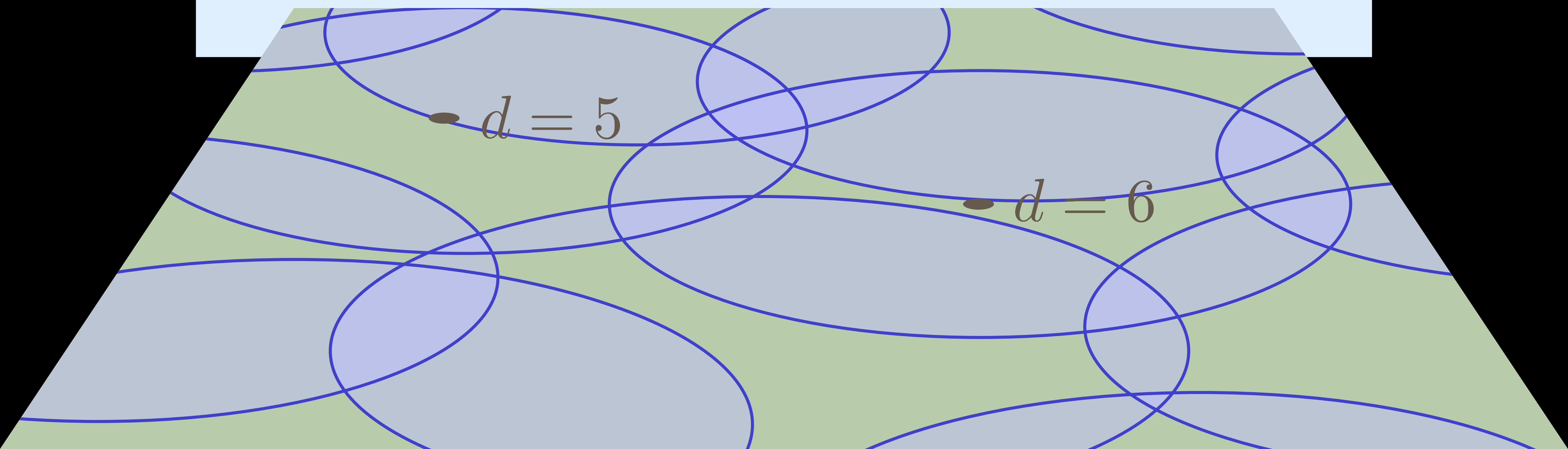
TOTAL DEGREE

- Degree of a region
  - Number of other regions is intersects
  - $d(R) = |\{R' : R \cap R \neq \emptyset\}|$
- Total degree of a set of regions



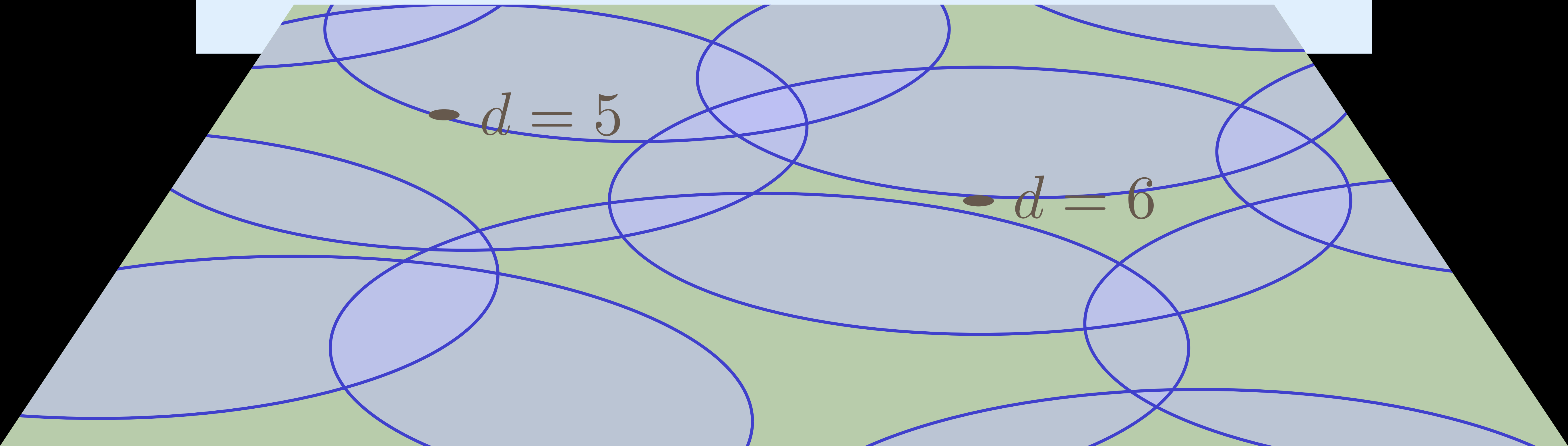
TOTAL DEGREE

- Degree of a region
  - Number of other regions is intersects
  - $d(R) = |\{R' : R \cap R \neq \emptyset\}|$
- Total degree of a set of regions
  - Sum of degrees of all regions



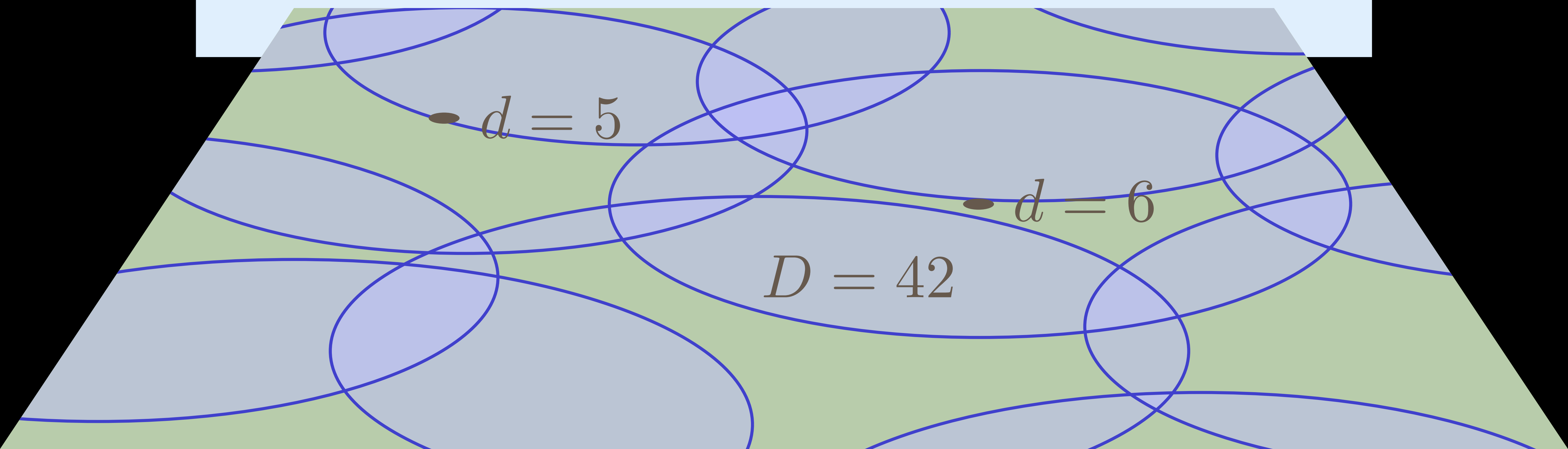
TOTAL DEGREE

- Degree of a region
  - Number of other regions is intersects
  - $d(R) = |\{R' : R \cap R \neq \emptyset\}|$
- Total degree of a set of regions
  - Sum of degrees of all regions
  - $D = \sum_R d(R)$



TOTAL DEGREE

- Degree of a region
  - Number of other regions is intersects
  - $d(R) = |\{R' : R \cap R \neq \emptyset\}|$
- Total degree of a set of regions
  - Sum of degrees of all regions
  - $D = \sum_R d(R)$



TOTAL DEGREE

Degree of a region

Number of other regions is intersects

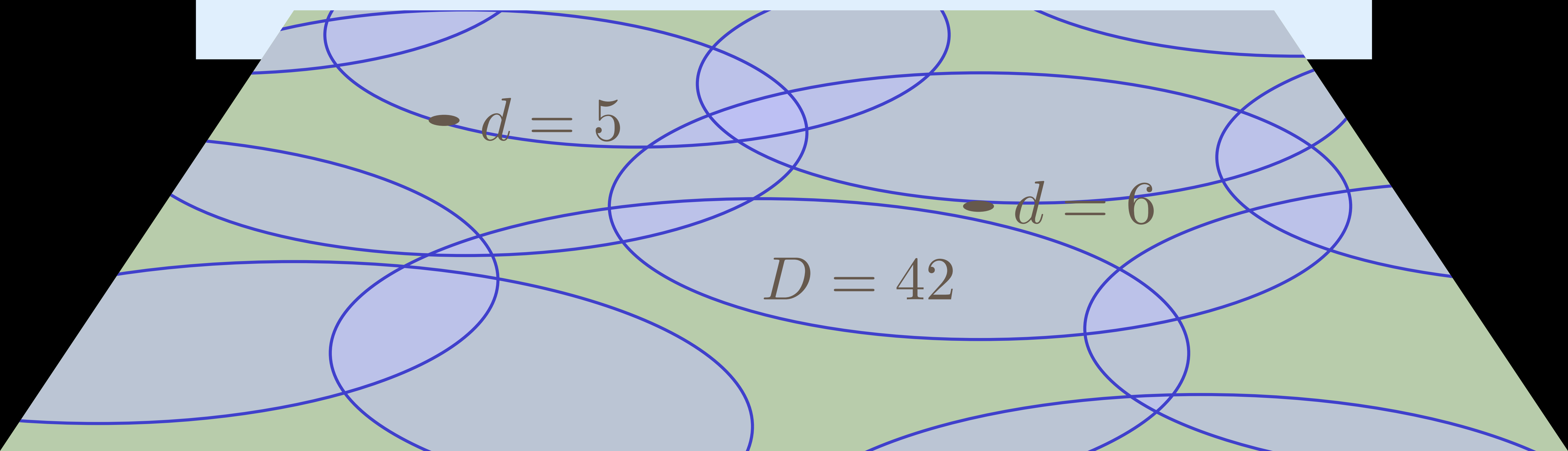
$d(R) = |\{R' : R \cap R \neq \emptyset\}|$

Total degree of a set of regions

Sum of degrees of all regions

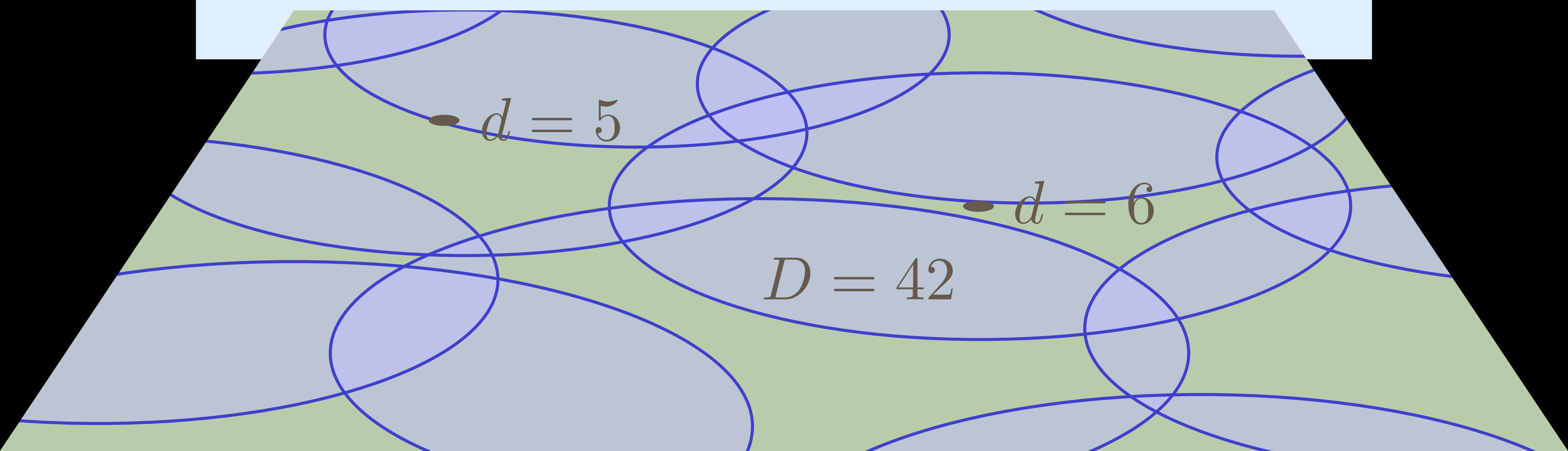
$D = \sum_R d(R)$

twice the number of edges in the disk intersection graph



TOTAL DEGREE

- Degree of a region
  - Number of other regions is intersects
  - $d(R) = |\{R' : R \cap R \neq \emptyset\}|$
- Total degree of a set of regions
  - Sum of degrees of all regions
  - $D = \sum_R d(R)$
  - twice the number of edges in the disk intersection graph
- Total degree is a *global* congestion measure.



PROBLEM STATEMENT

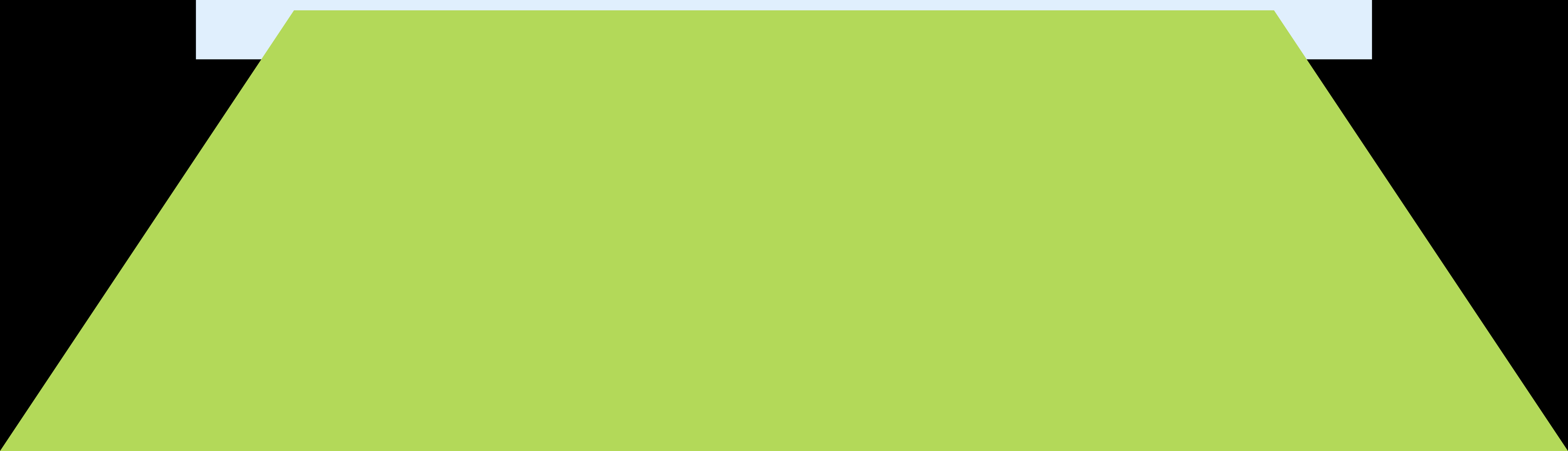
PROBLEM STATEMENT

Input

PROBLEM STATEMENT

Input

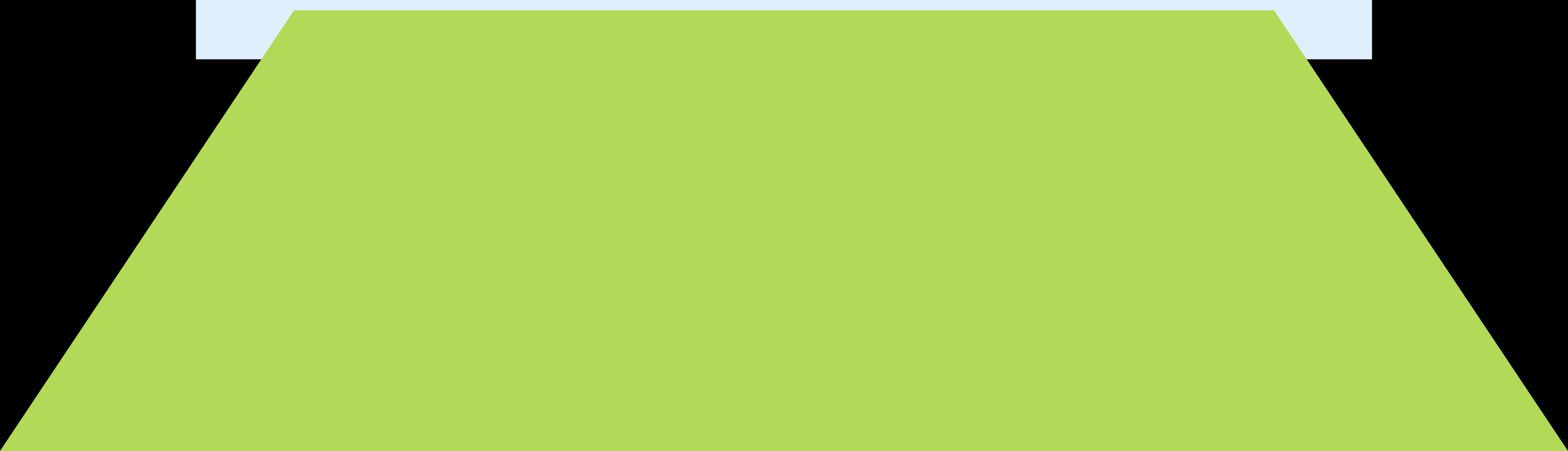
Set of n moving points with initial regions



PROBLEM STATEMENT

Input

- Set of n moving points with initial regions
- Target value τ

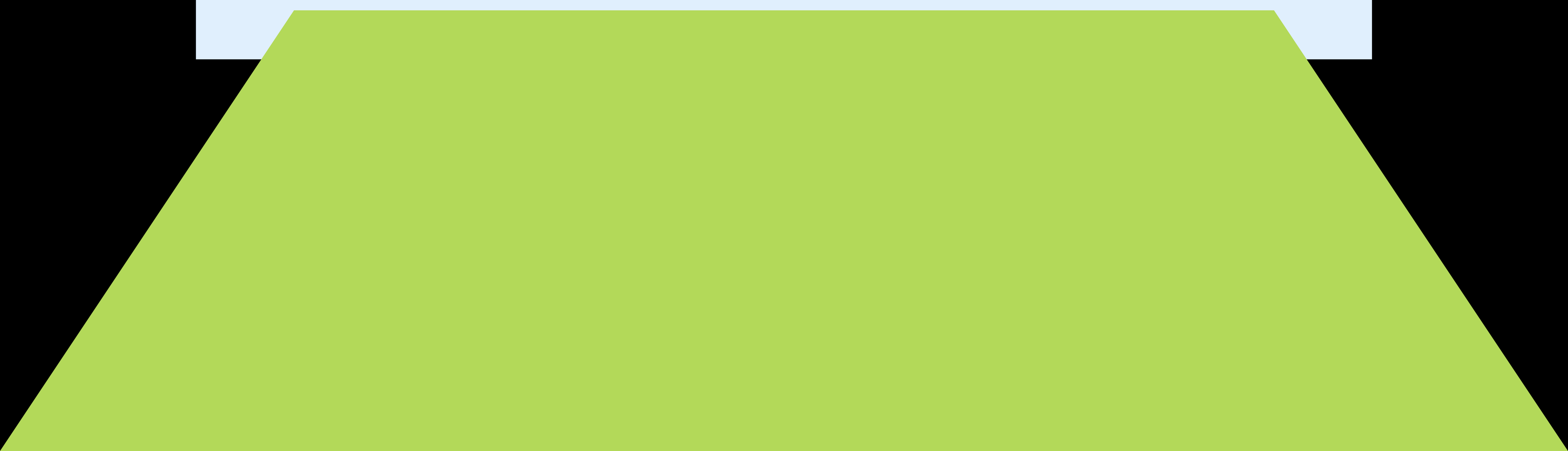


PROBLEM STATEMENT

Input

- Set of n moving points with initial regions
- Target value τ

Strategy



PROBLEM STATEMENT

Input

- Set of n moving points with initial regions
- Target value τ

Strategy

- Select one region to query each time step

PROBLEM STATEMENT

Input

- Set of n moving points with initial regions
- Target value τ

Strategy

- Select one region to query each time step
- Keep the total degree of the intersection graph under τ

PROBLEM STATEMENT

Input

- Set of n moving points with initial regions
- Target value τ

Strategy

- Select one region to query each time step
- Keep the total degree of the intersection graph under τ

Problem

PROBLEM STATEMENT

Input

Set of n moving points with initial regions
Target value τ

Strategy

Select one region to query each time step
Keep the total degree of the intersection graph under τ

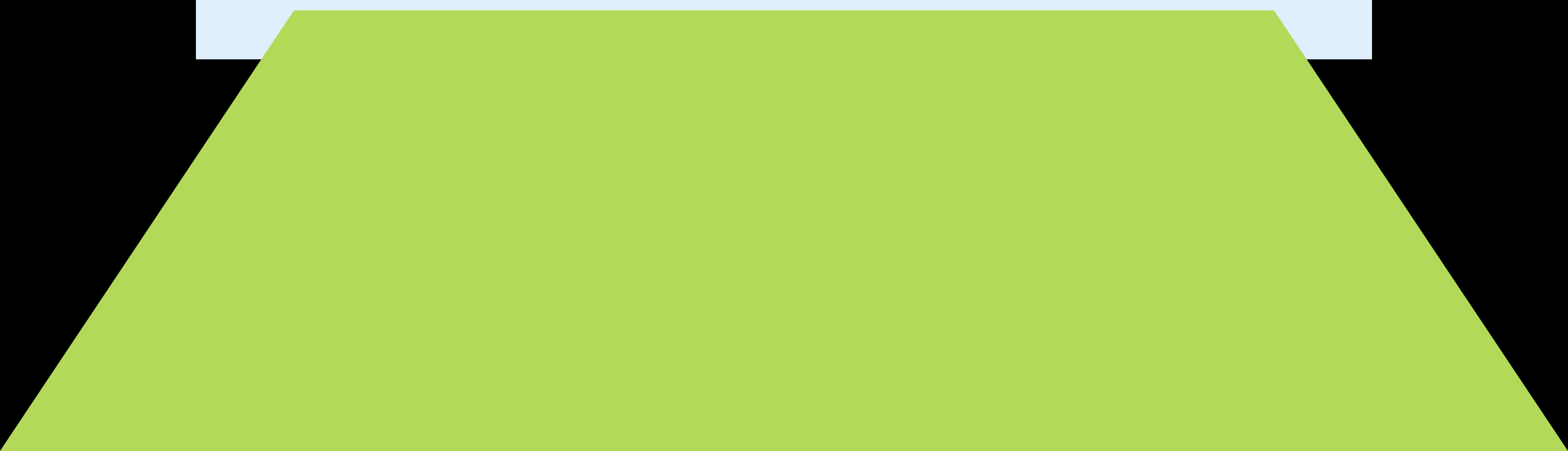
Problem

Decide whether such a strategy exists.

STATIC MOVING POINTS

STATIC MOVING POINTS

Consider n non-moving points



STATIC MOVING POINTS

Consider n non-moving points

STATIC MOVING POINTS

- Consider n non-moving points
  - Suppose that for each point we know a *location* p_i and an associated *time stamp* t_i

STATIC MOVING POINTS

Consider n non-moving points

Suppose that for each point we know a *location* p_i and an associated *time stamp* t_i

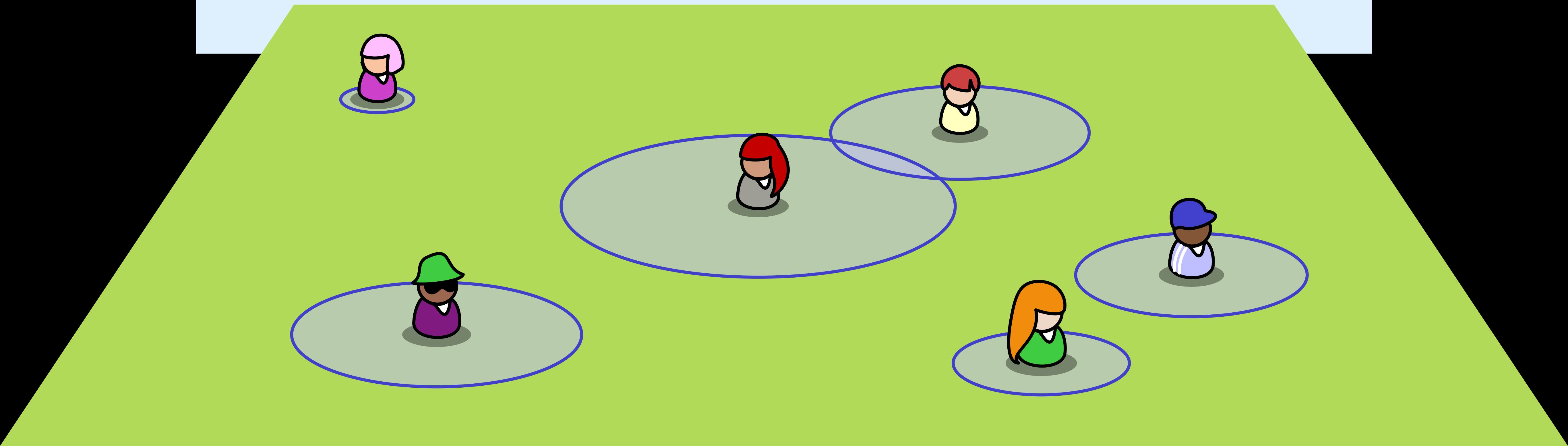
Then the current uncertainty region of each point is a disk centered at p_i

STATIC MOVING POINTS

Consider n non-moving points

Suppose that for each point we know a *location* p_i and an associated *time stamp* t_i

Then the current uncertainty region of each point is a disk centered at p_i



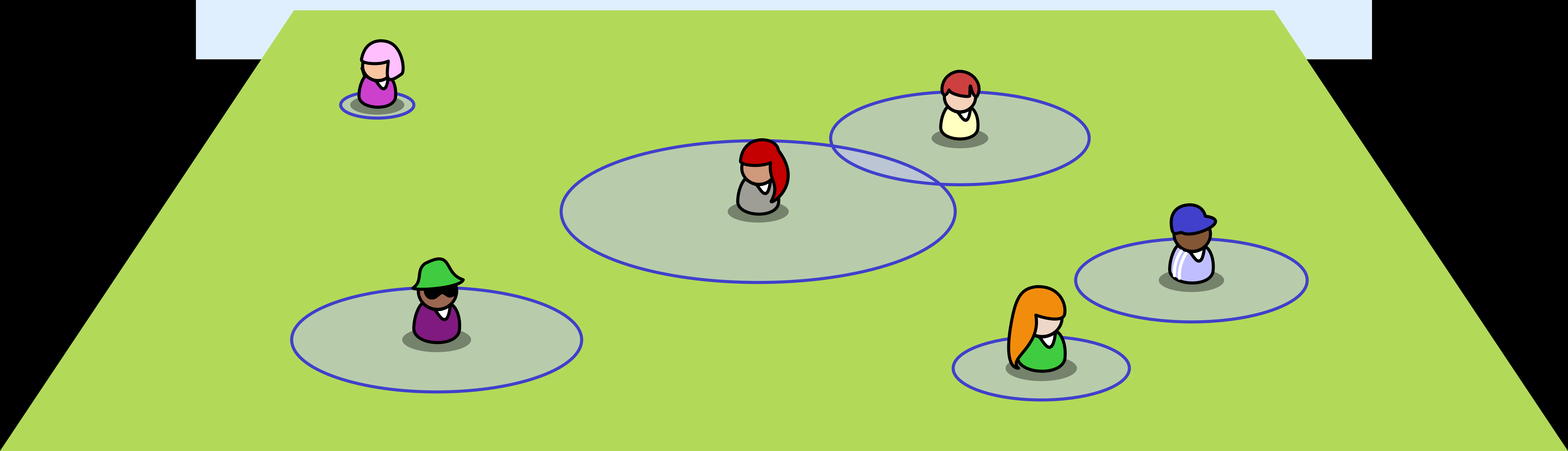
STATIC MOVING POINTS

Consider n non-moving points

Suppose that for each point we know a *location* p_i and an associated *time stamp* t_i

Then the current uncertainty region of each point is a disk centered at p_i

Each unit of time, all regions grow by d



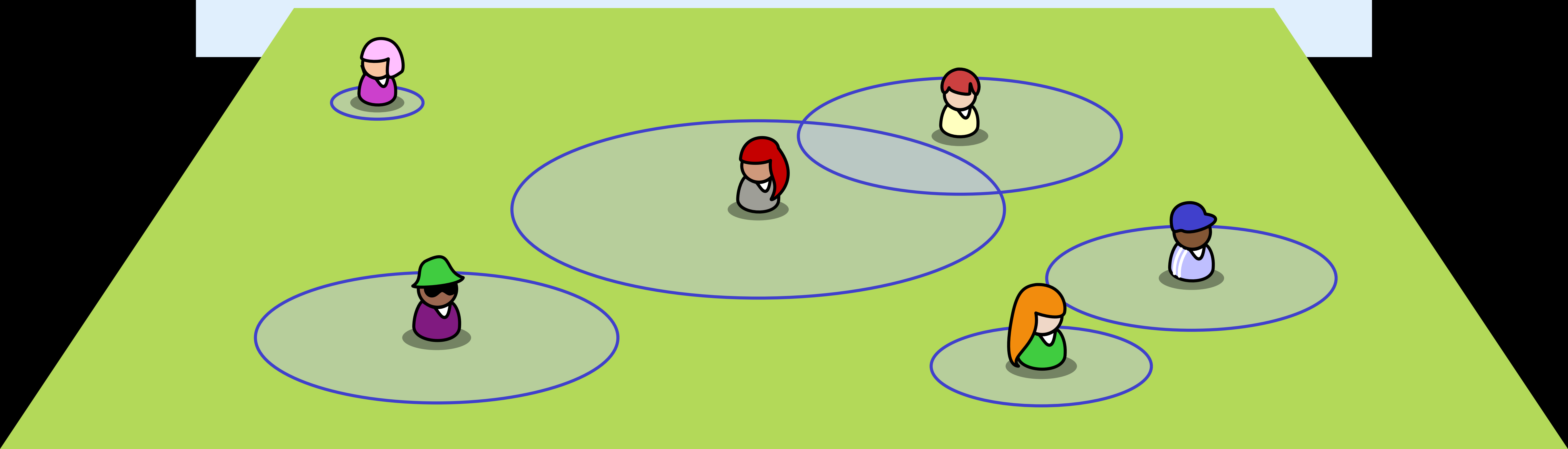
STATIC MOVING POINTS

Consider n non-moving points

Suppose that for each point we know a *location* p_i and an associated *time stamp* t_i

Then the current uncertainty region of each point is a disk centered at p_i

Each unit of time, all regions grow by d



STATIC MOVING POINTS

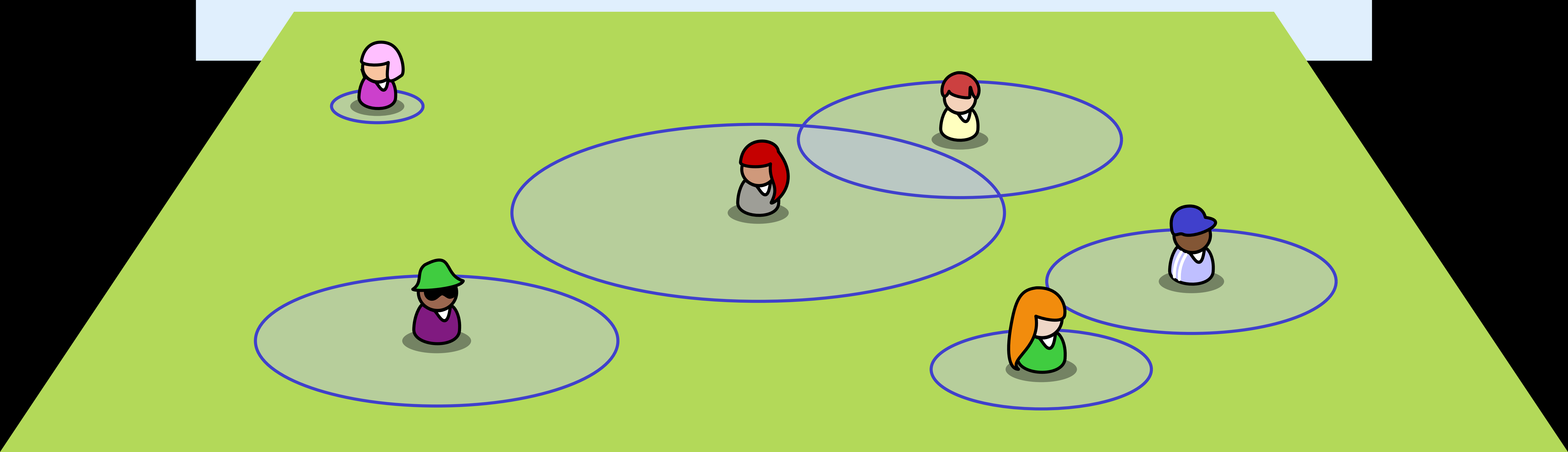
Consider n non-moving points

Suppose that for each point we know a *location* p_i and an associated *time stamp* t_i

Then the current uncertainty region of each point is a disk centered at p_i

Each unit of time, all regions grow by d

We can *query* a point to update it



STATIC MOVING POINTS

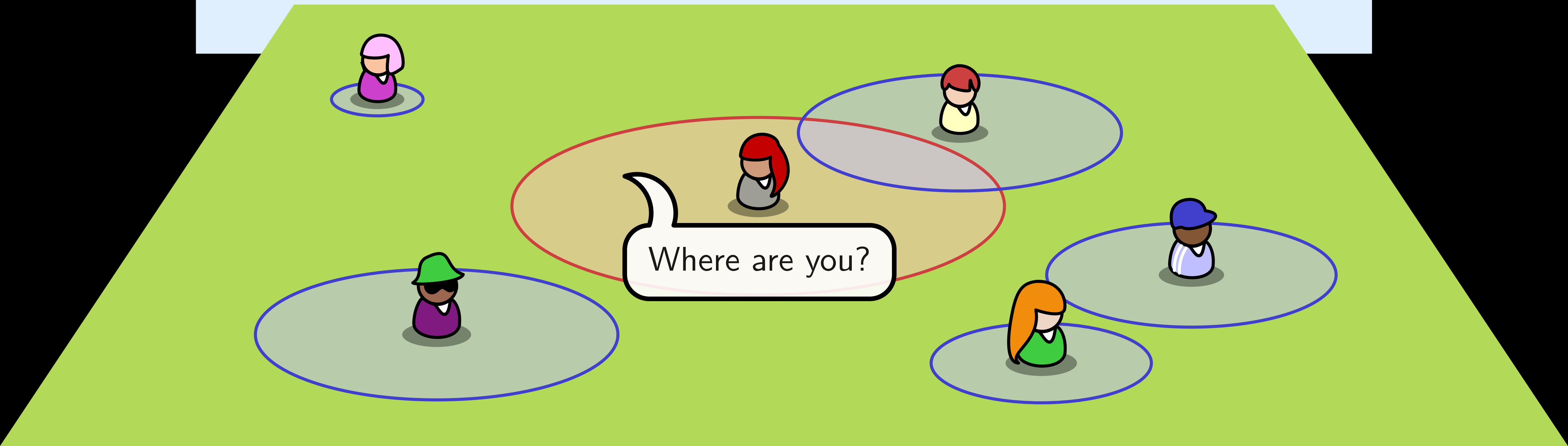
Consider n non-moving points

Suppose that for each point we know a *location* p_i and an associated *time stamp* t_i

Then the current uncertainty region of each point is a disk centered at p_i

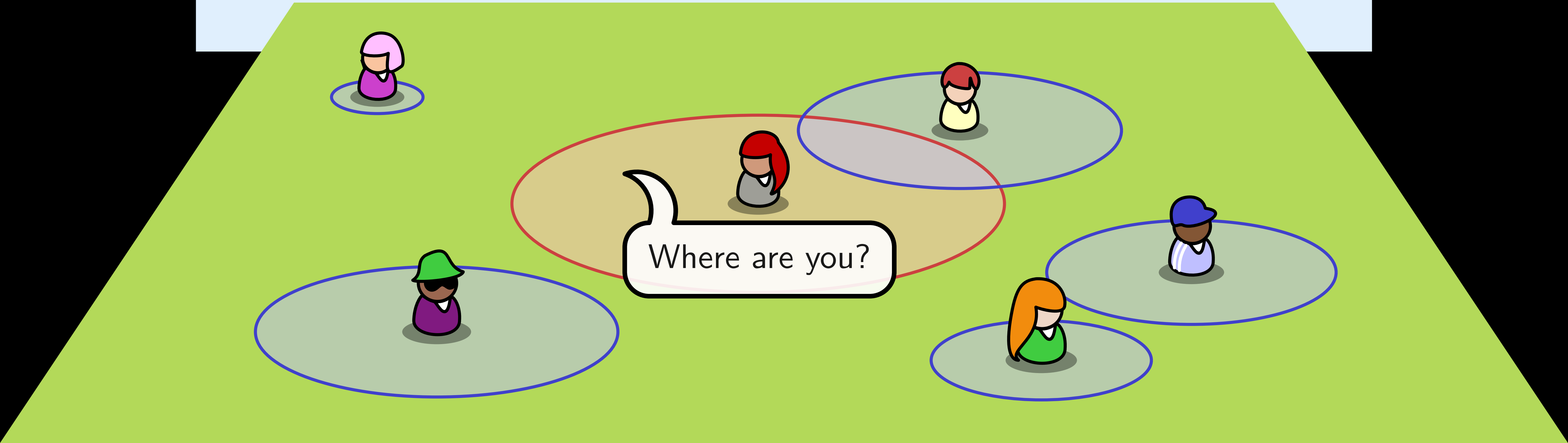
Each unit of time, all regions grow by d

We can *query* a point to update it



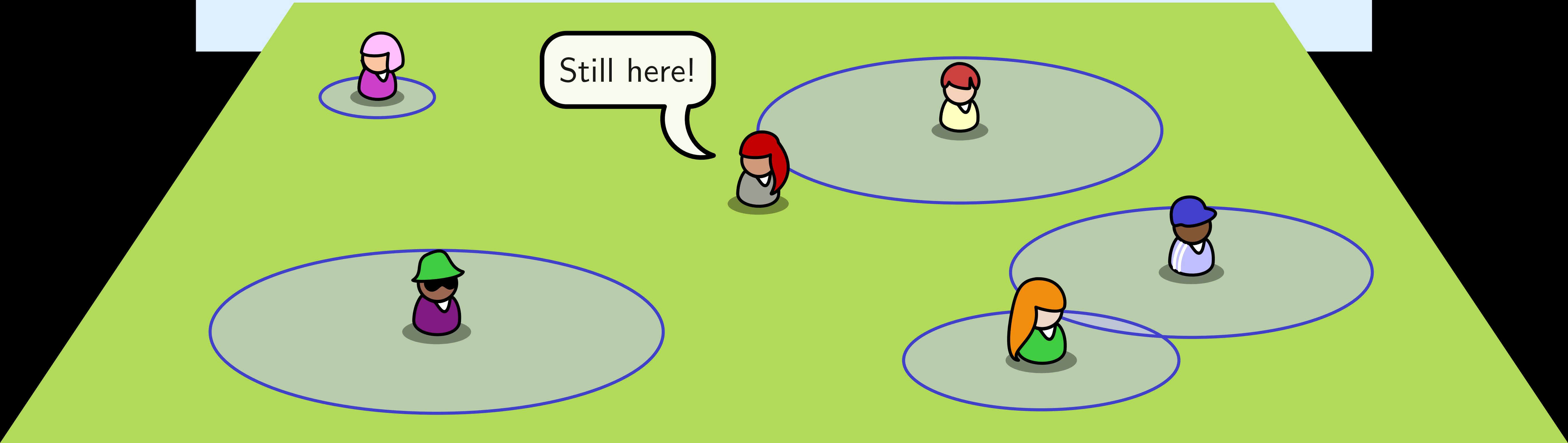
STATIC MOVING POINTS

- Consider n non-moving points
  - Suppose that for each point we know a *location* p_i and an associated *time stamp* t_i
  - Then the current uncertainty region of each point is a disk centered at p_i
  - Each unit of time, all regions grow by d
- We can *query* a point to update it
  - Takes 1 unit of time to execute



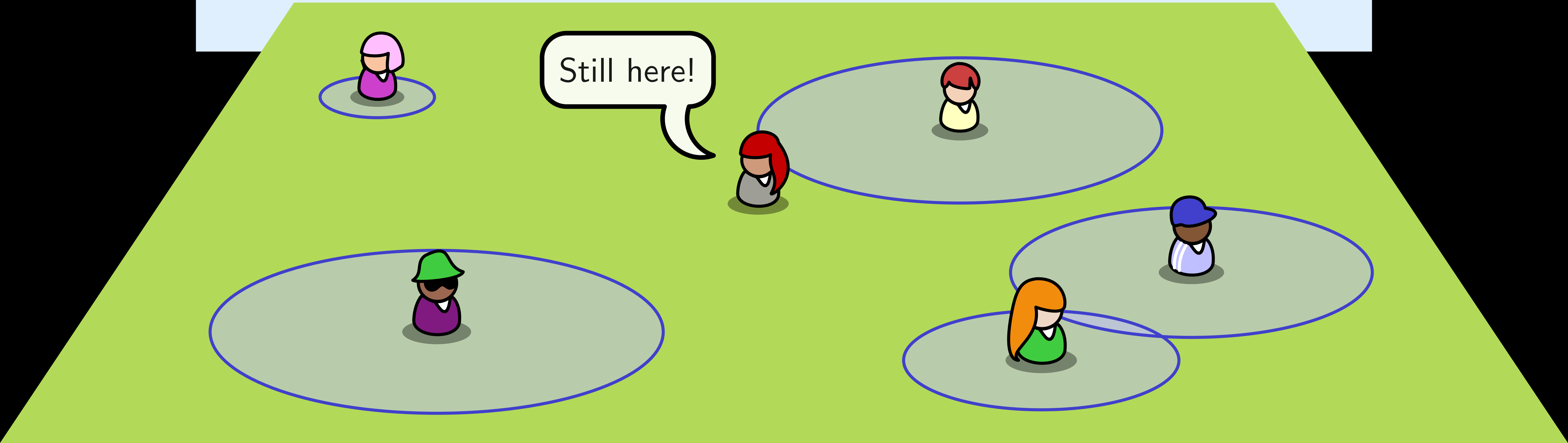
STATIC MOVING POINTS

- Consider n non-moving points
  - Suppose that for each point we know a *location* p_i and an associated *time stamp* t_i
  - Then the current uncertainty region of each point is a disk centered at p_i
  - Each unit of time, all regions grow by d
- We can *query* a point to update it
  - Takes 1 unit of time to execute



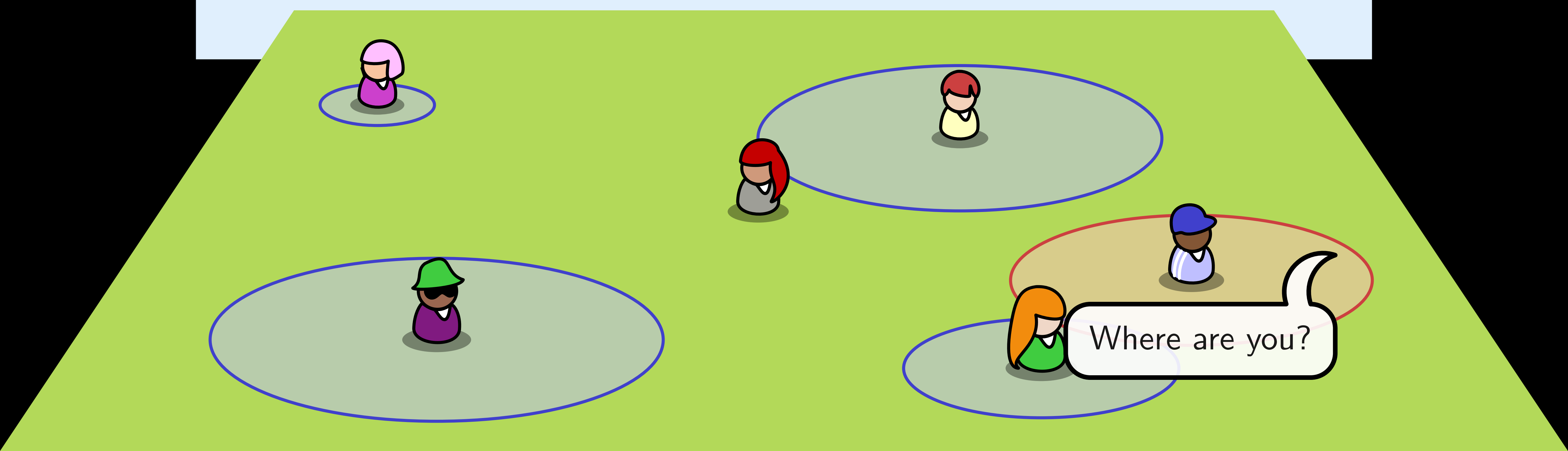
STATIC MOVING POINTS

- Consider n non-moving points
  - Suppose that for each point we know a *location* p_i and an associated *time stamp* t_i
  - Then the current uncertainty region of each point is a disk centered at p_i
  - Each unit of time, all regions grow by d
- We can *query* a point to update it
  - Takes 1 unit of time to execute
  - After a query, one region collapses to a point, but all other regions grow



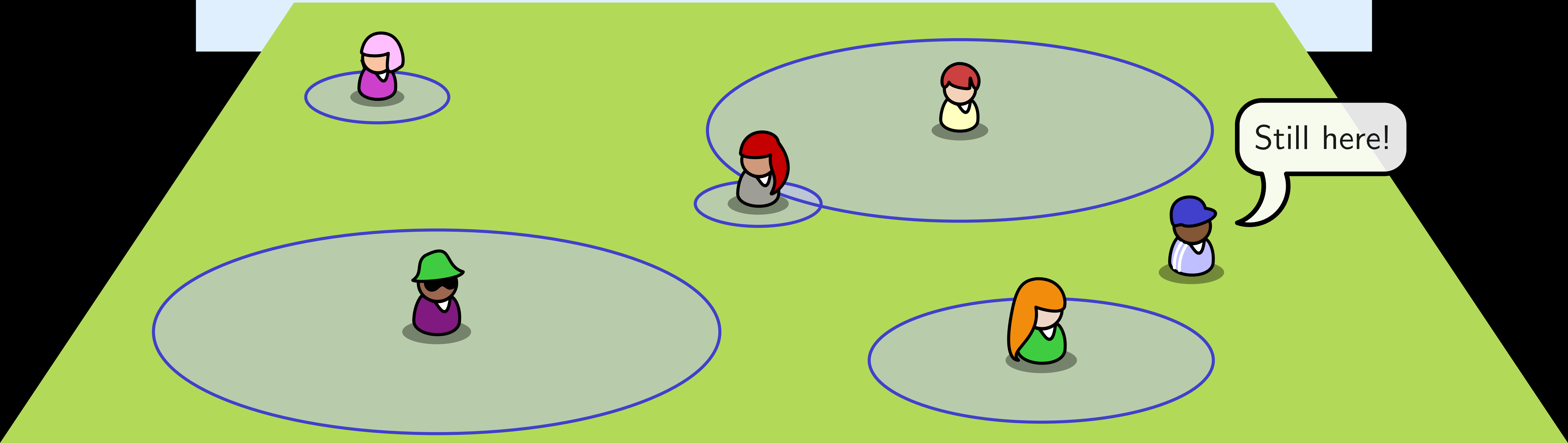
STATIC MOVING POINTS

- Consider n non-moving points
  - Suppose that for each point we know a *location* p_i and an associated *time stamp* t_i
  - Then the current uncertainty region of each point is a disk centered at p_i
  - Each unit of time, all regions grow by d
- We can *query* a point to update it
  - Takes 1 unit of time to execute
  - After a query, one region collapses to a point, but all other regions grow



STATIC MOVING POINTS

- Consider n non-moving points
  - Suppose that for each point we know a *location* p_i and an associated *time stamp* t_i
  - Then the current uncertainty region of each point is a disk centered at p_i
  - Each unit of time, all regions grow by d
- We can *query* a point to update it
  - Takes 1 unit of time to execute
  - After a query, one region collapses to a point, but all other regions grow



1-DIMENSIONAL POINTS

1-DIMENSIONAL POINTS

Points are on a line

1-DIMENSIONAL POINTS

Points are on a line

1-DIMENSIONAL POINTS

Points are on a line

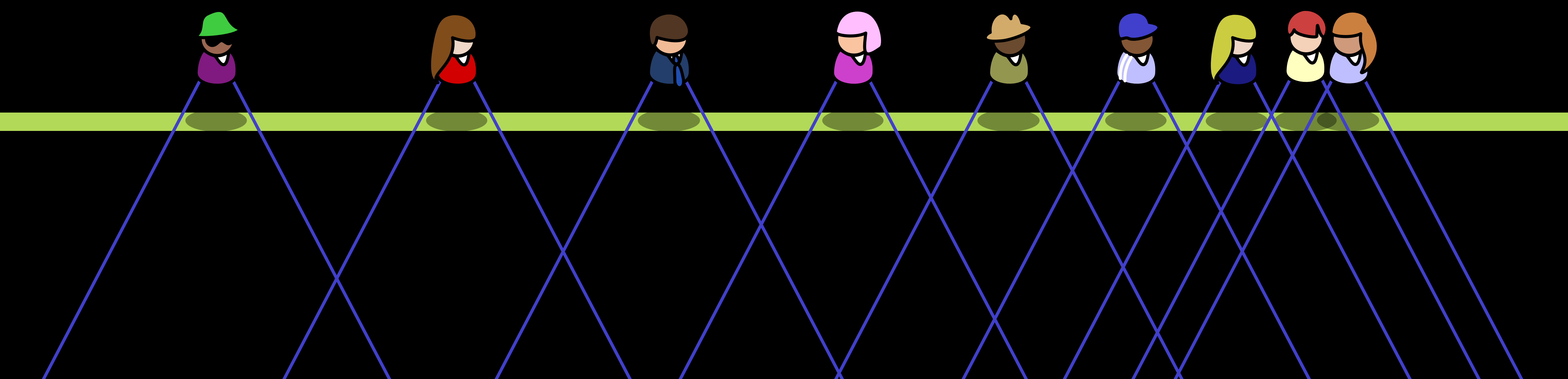
1-DIMENSIONAL POINTS

- Points are on a line
- We play the same game

1-DIMENSIONAL POINTS

Points are on a line

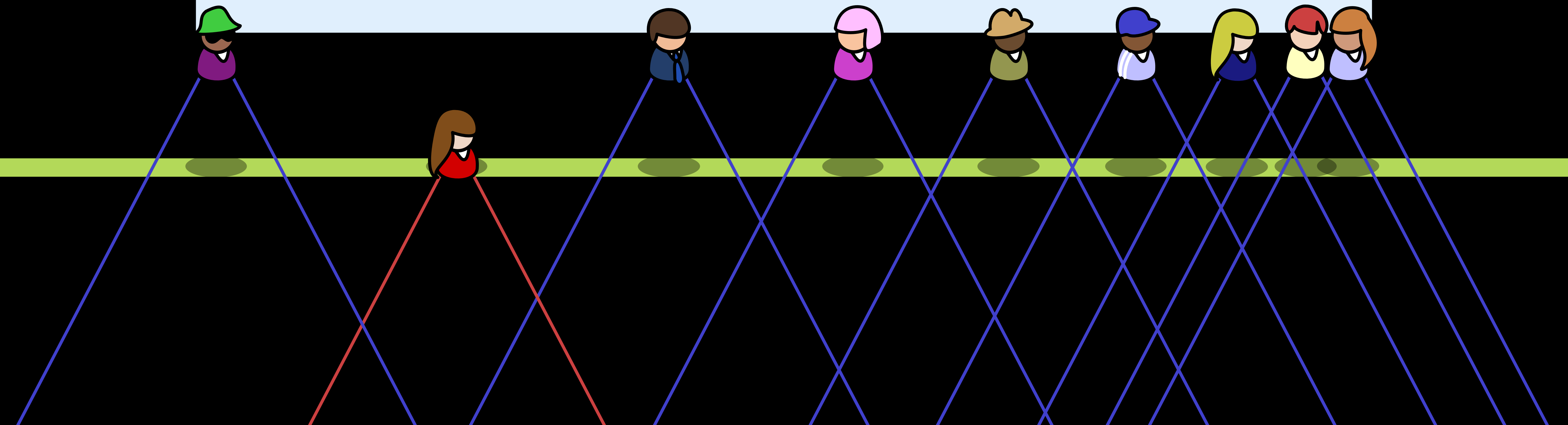
We play the same game



1-DIMENSIONAL POINTS

Points are on a line

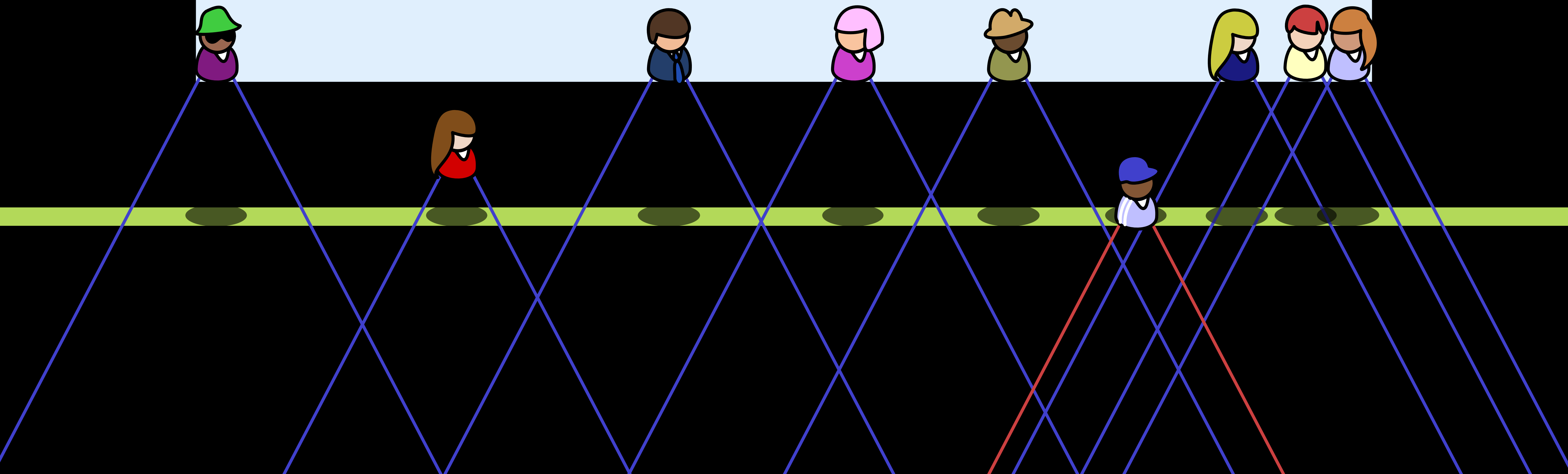
We play the same game



1-DIMENSIONAL POINTS

Points are on a line

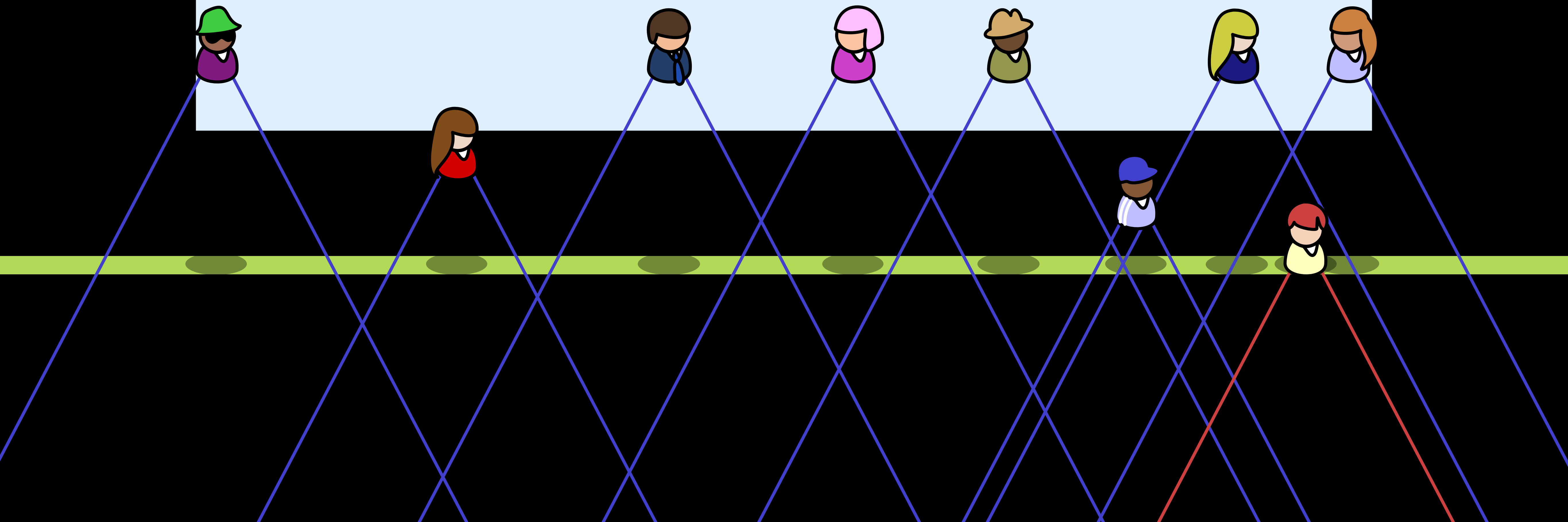
We play the same game



1-DIMENSIONAL POINTS

Points are on a line

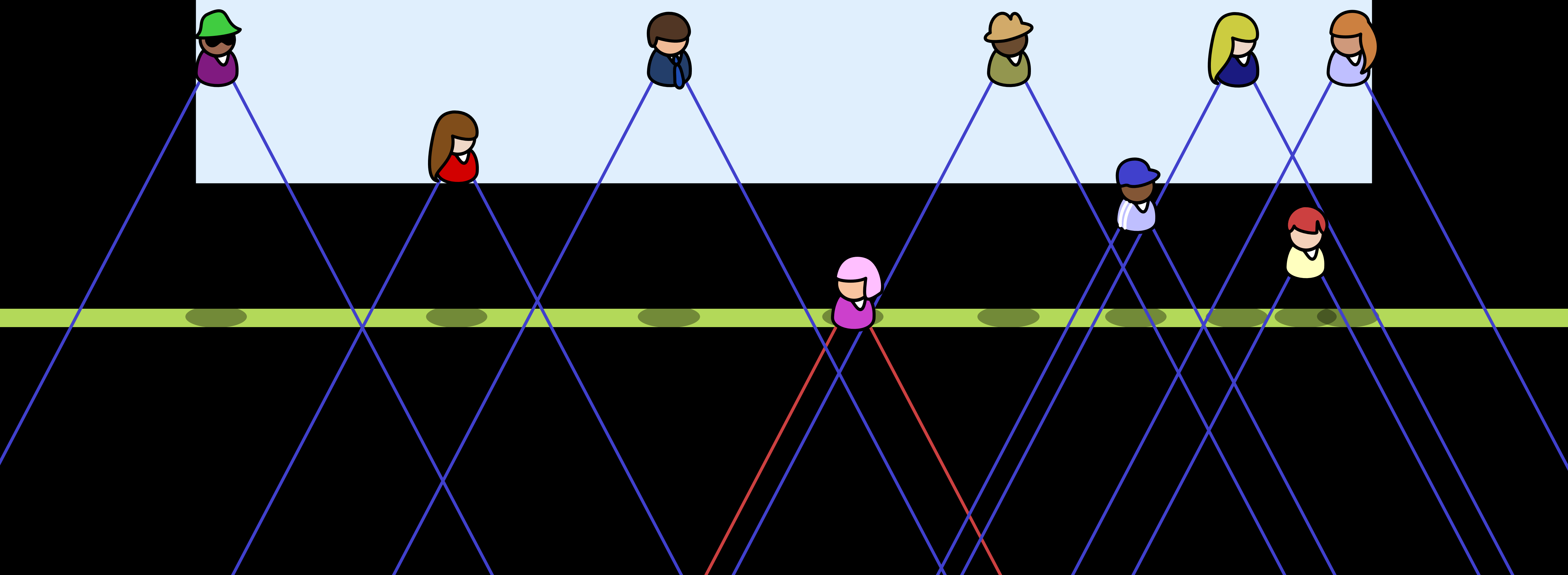
We play the same game



1-DIMENSIONAL POINTS

Points are on a line

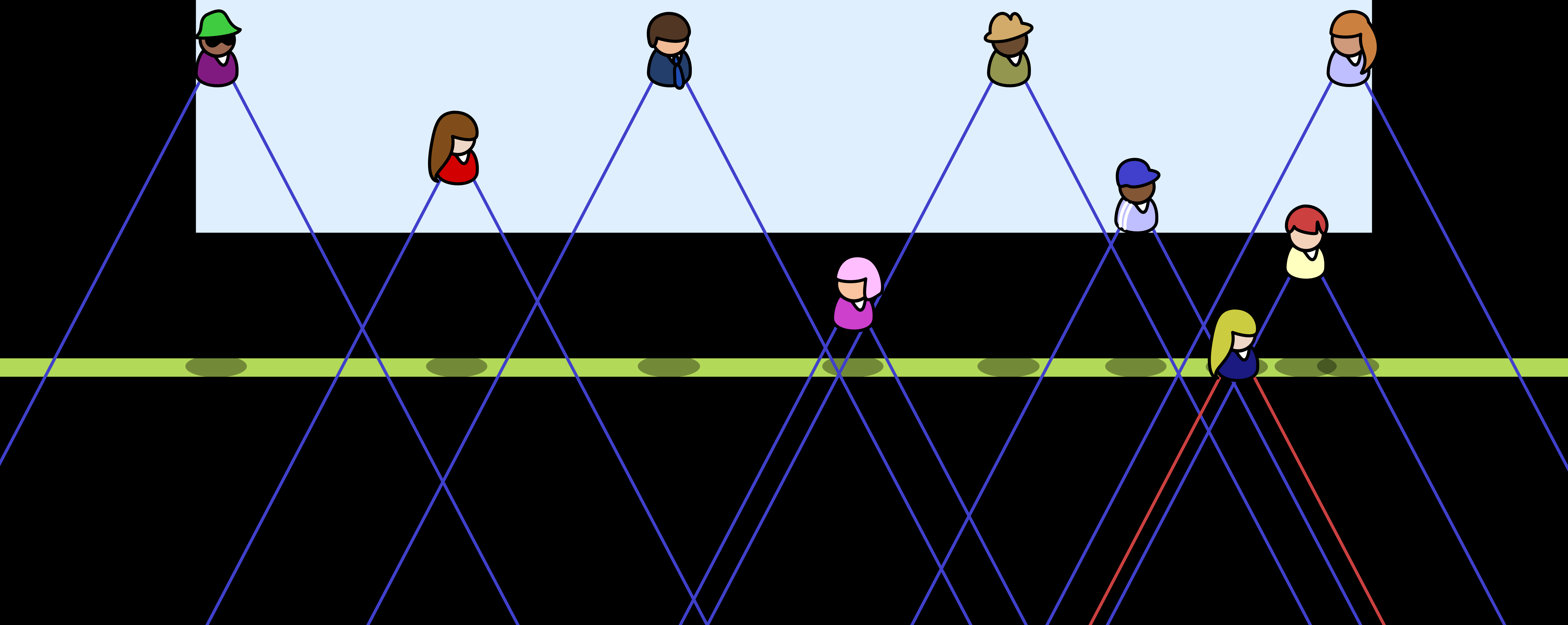
We play the same game



1-DIMENSIONAL POINTS

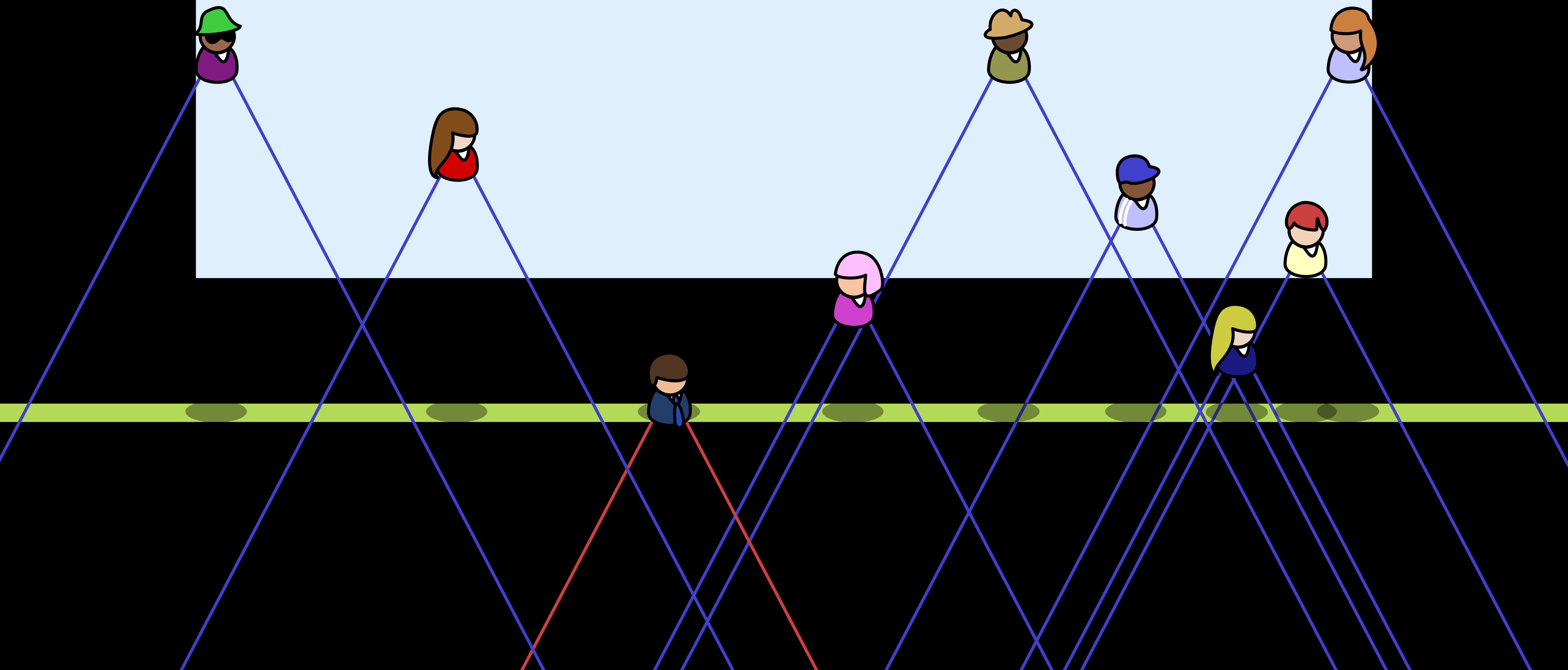
Points are on a line

We play the same game



1-DIMENSIONAL POINTS

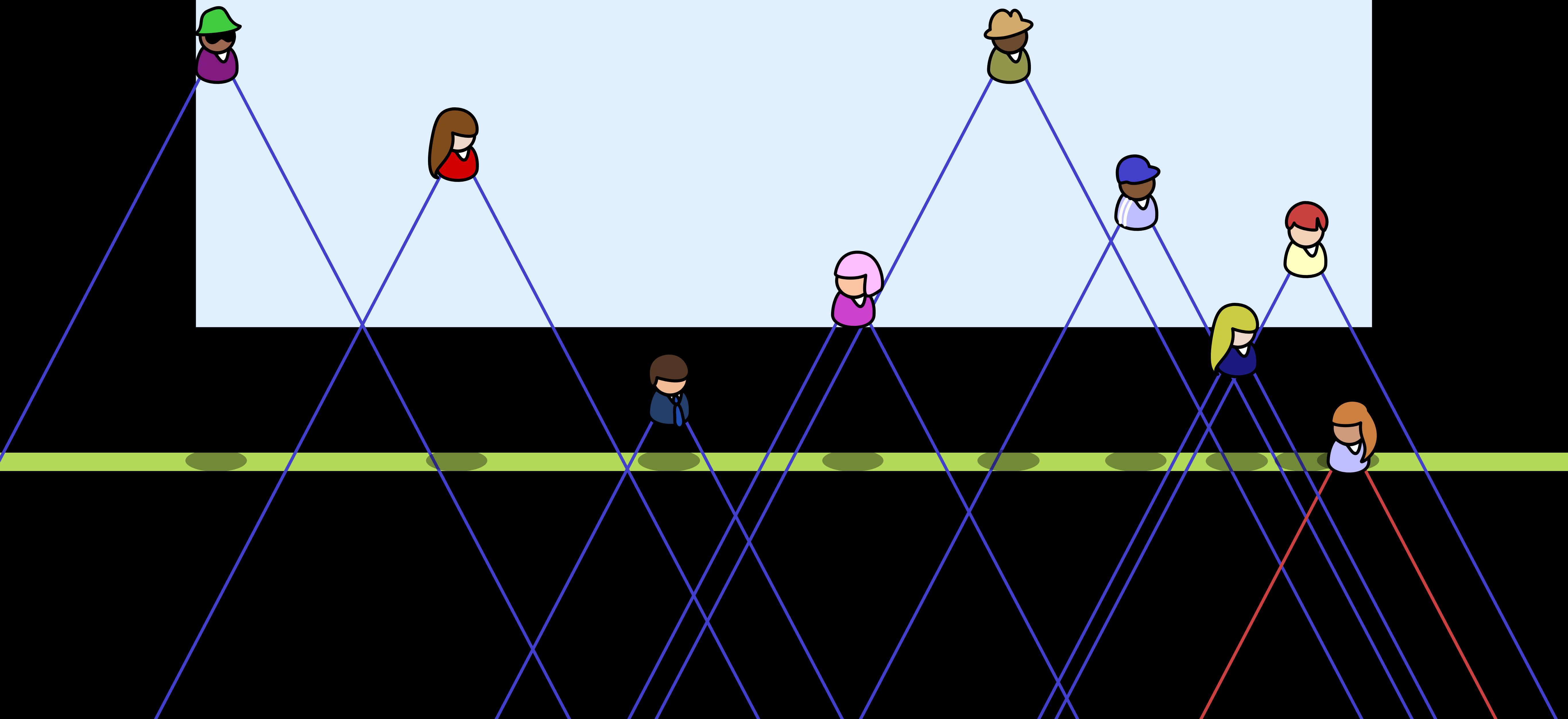
- Points are on a line
- We play the same game



1-DIMENSIONAL POINTS

Points are on a line

We play the same game

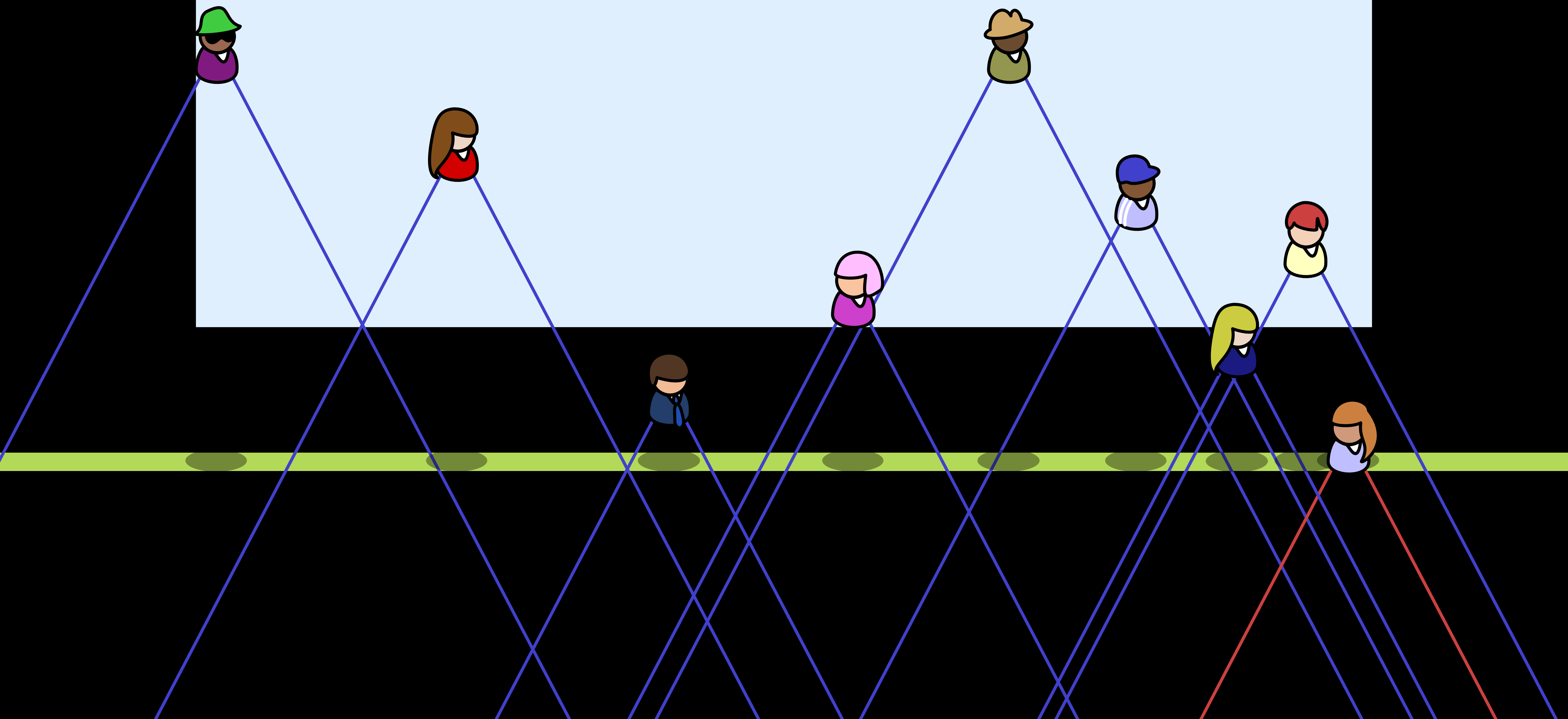


1-DIMENSIONAL POINTS

Points are on a line

We play the same game

We want a small intersection graph

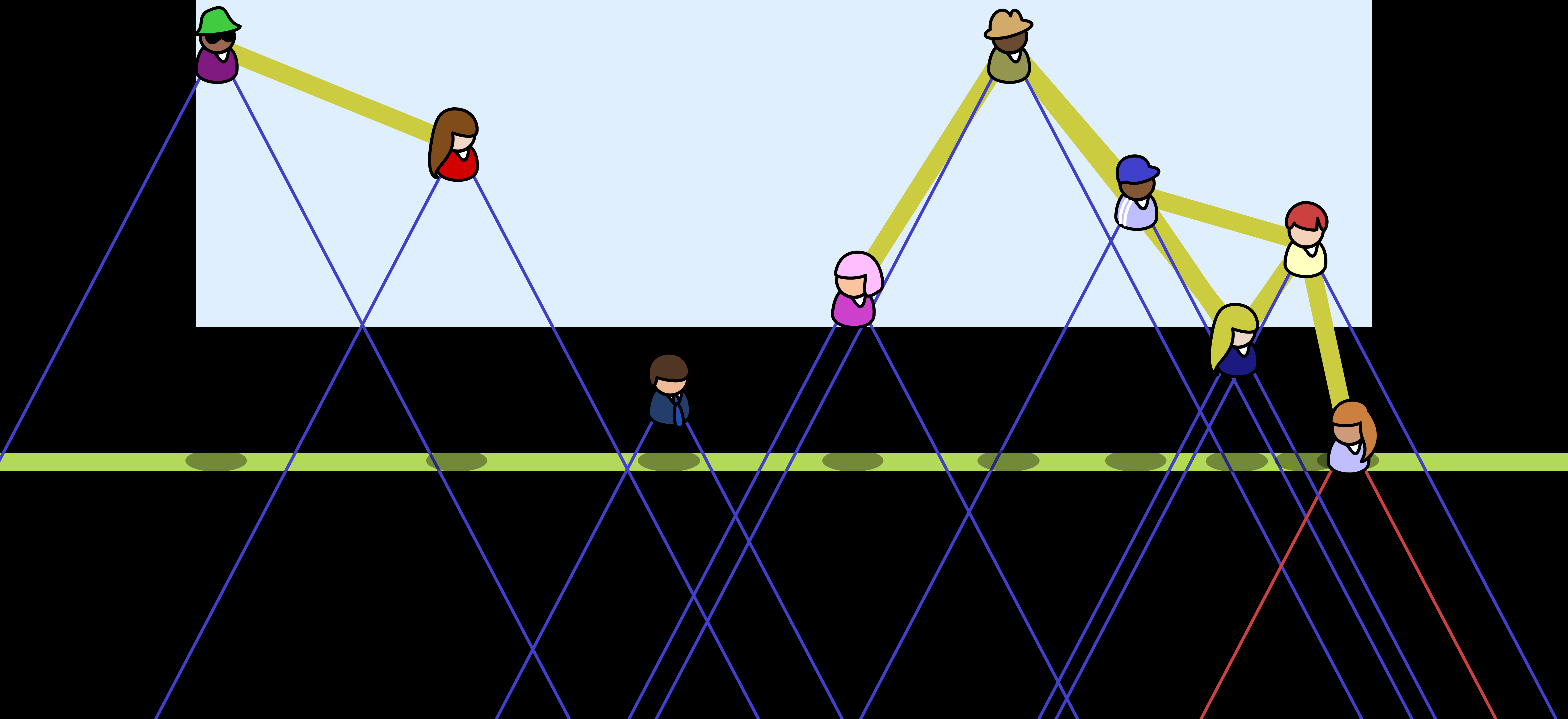


1-DIMENSIONAL POINTS

Points are on a line

We play the same game

We want a small intersection graph

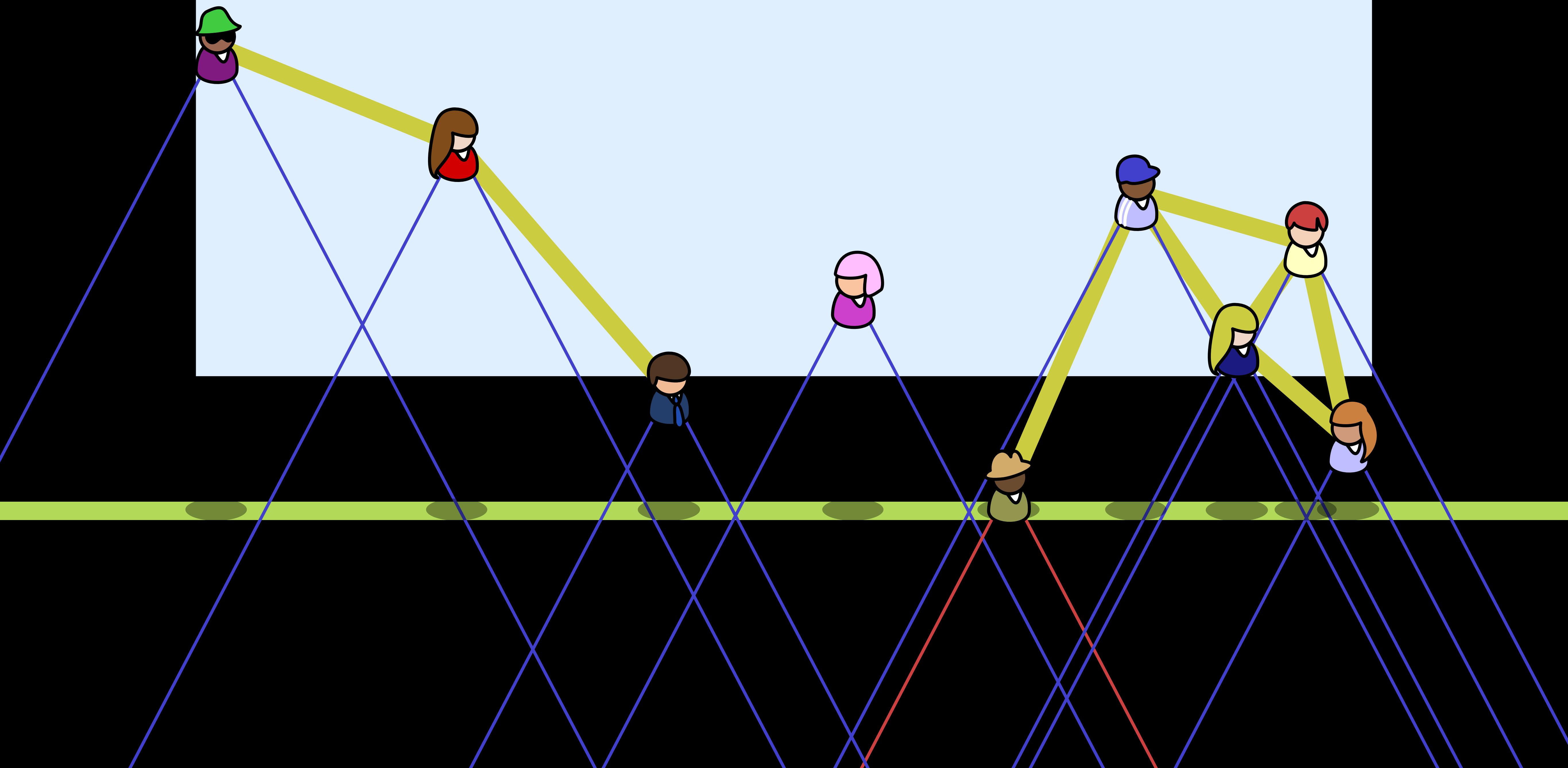


1-DIMENSIONAL POINTS

Points are on a line

We play the same game

We want a small intersection graph



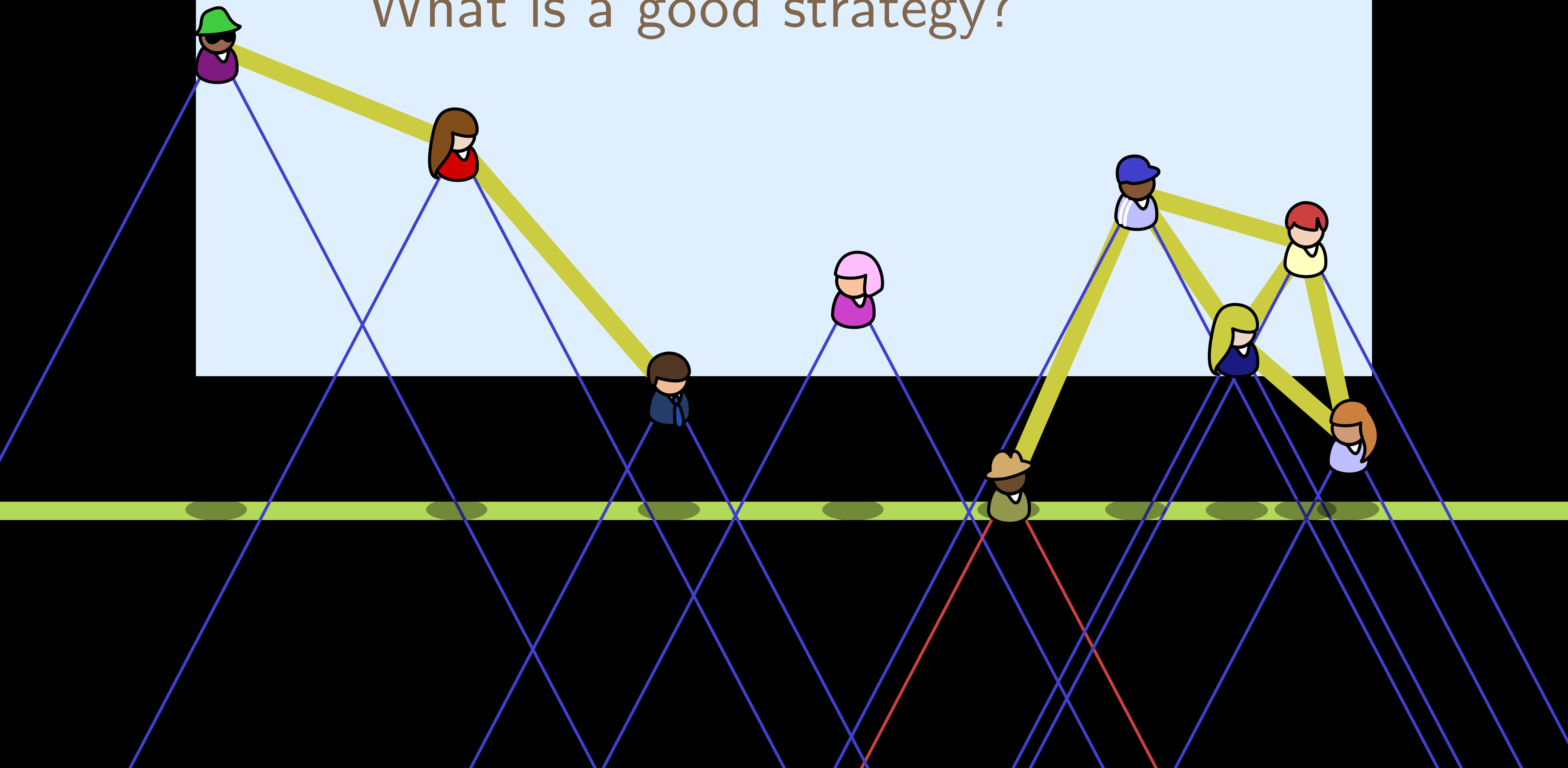
1-DIMENSIONAL POINTS

Points are on a line

We play the same game

We want a small intersection graph

What is a good strategy?



THEOREM

THEOREM

For any target value τ, either
(i) any query strategy has uncertainty intervals with intersection graph of degree $\Omega(\tau)$ at some point in any time interval of length τ, or
(ii) a simple query strategy guarantees that the total degree in the intersection graph of the uncertainty intervals is $O(\tau)$ at all times.

THEOREM

For any target value τ, either
(i) ..., or
(ii) a simple query strategy guarantees that the total degree in the intersection graph of the uncertainty intervals is $O(\tau)$ at all times.

> The *smallest* τ for which (ii) applies is the *critical* degree

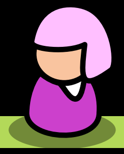
THE STRATEGY

THE STRATEGY

Critical radius of point p

THE STRATEGY

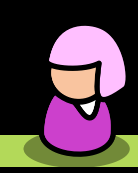
Critical radius of point p



THE STRATEGY

Critical radius of point p

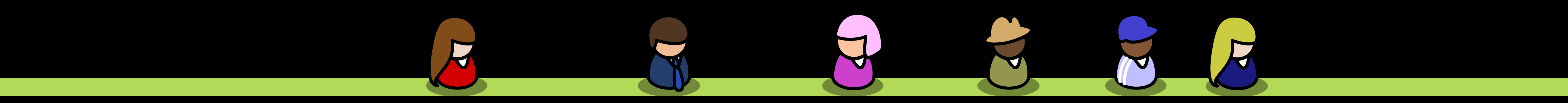
Smallest r such that the ball of radius r around p contains at least $\frac{c\tau}{r}$ other points



THE STRATEGY

Critical radius of point p

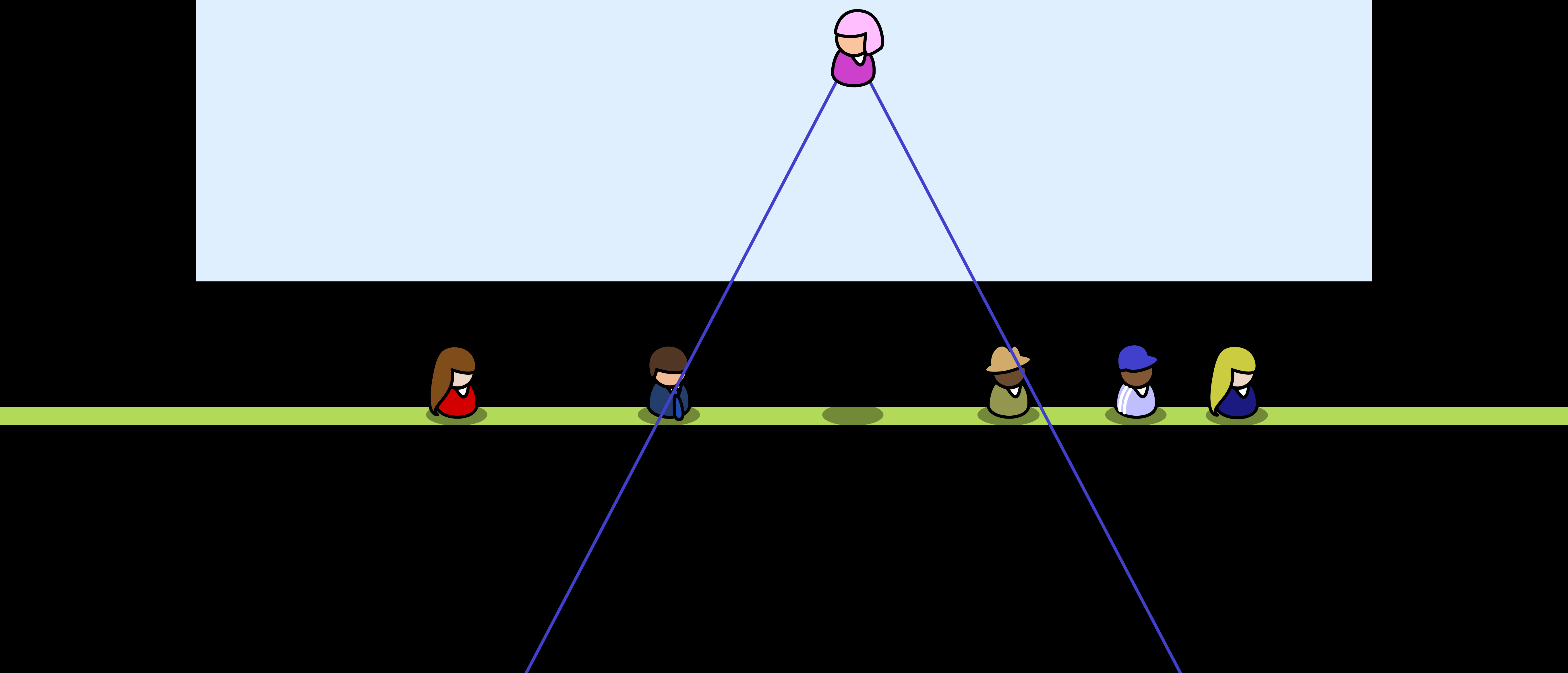
Smallest r such that the ball of radius r around p contains at least $\frac{c\tau}{r}$ other points



THE STRATEGY

Critical radius of point p

Smallest r such that the ball of radius r around p contains at least $\frac{c\tau}{r}$ other points

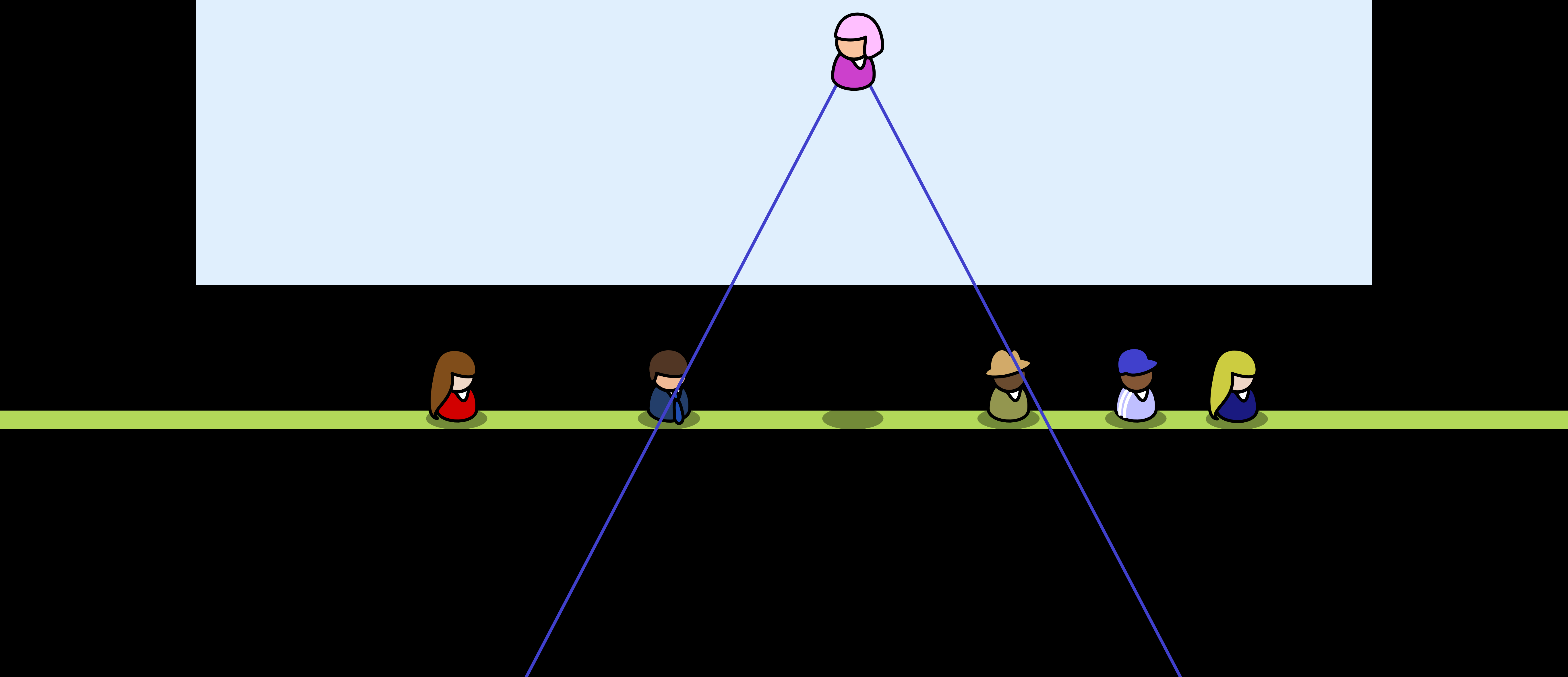


THE STRATEGY

Critical radius of point p

Smallest r such that the ball of radius r around p contains at least $\frac{c\tau}{r}$ other points

Points in denser areas have smaller r

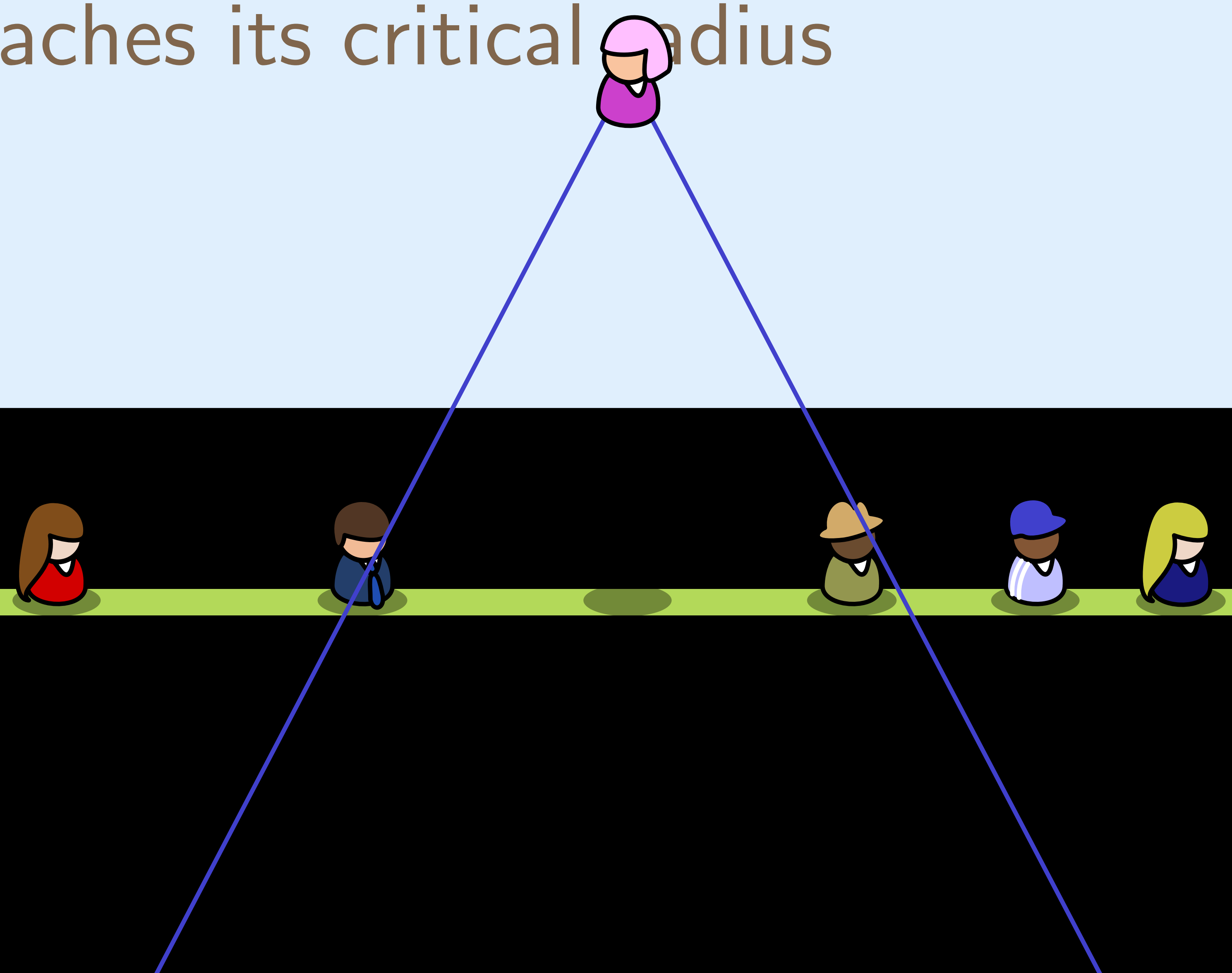


THE STRATEGY

Critical radius of point p

- Smallest r such that the ball of radius r around p contains at least $\frac{c\tau}{r}$ other points
- Points in denser areas have smaller r

Plan: query each point before it reaches its critical radius



THE STRATEGY

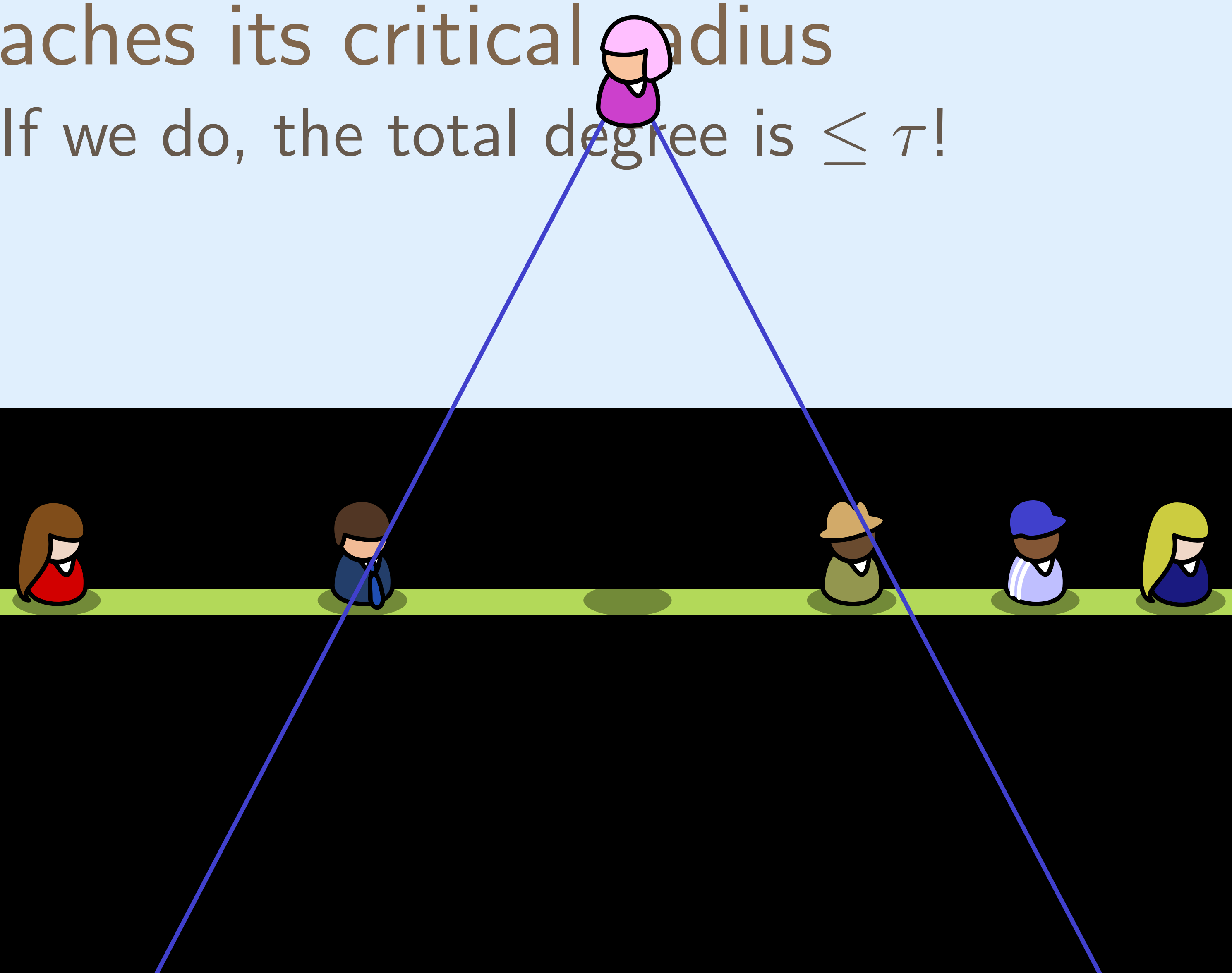
Critical radius of point p

Smallest r such that the ball of radius r around p contains at least $\frac{c\tau}{r}$ other points

Points in denser areas have smaller r

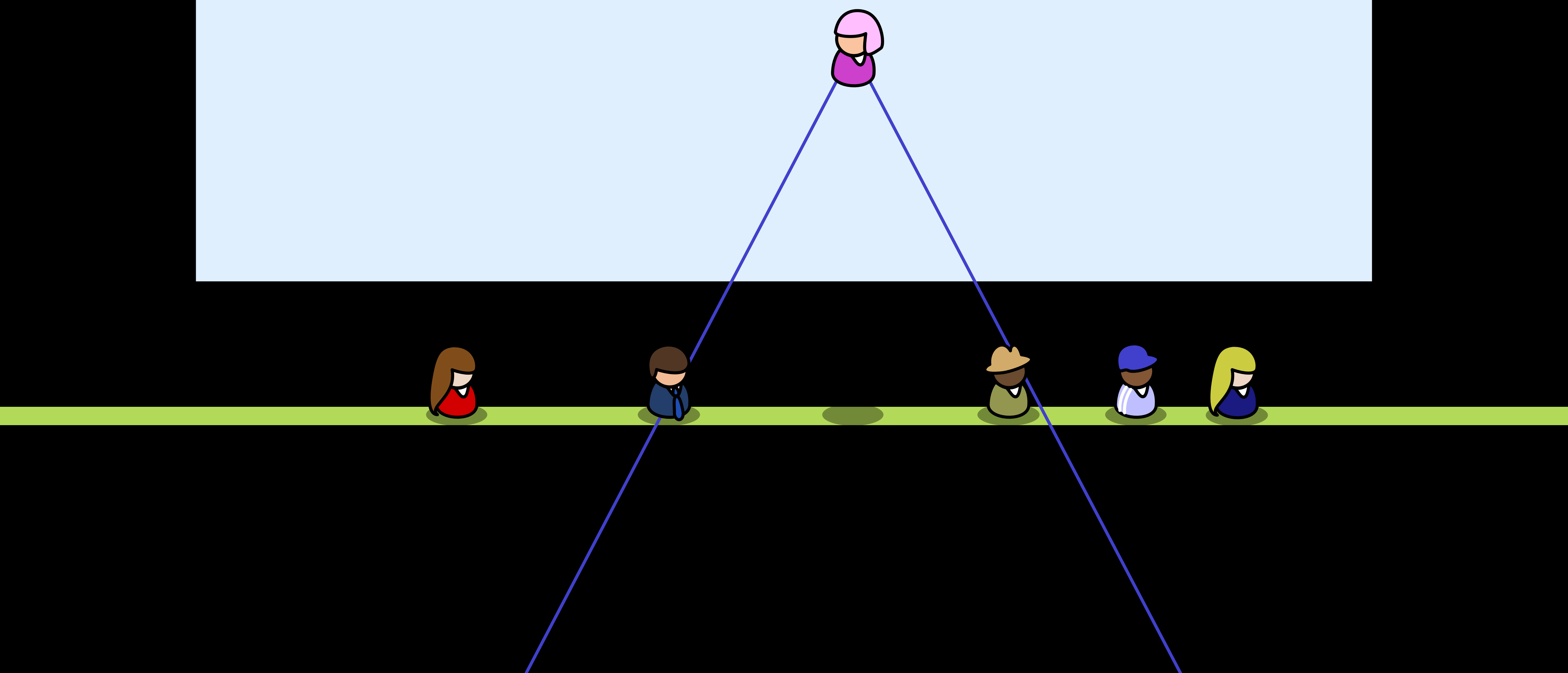
Plan: query each point before it reaches its critical radius

If we do, the total degree is $\leq \tau$!



THE STRATEGY

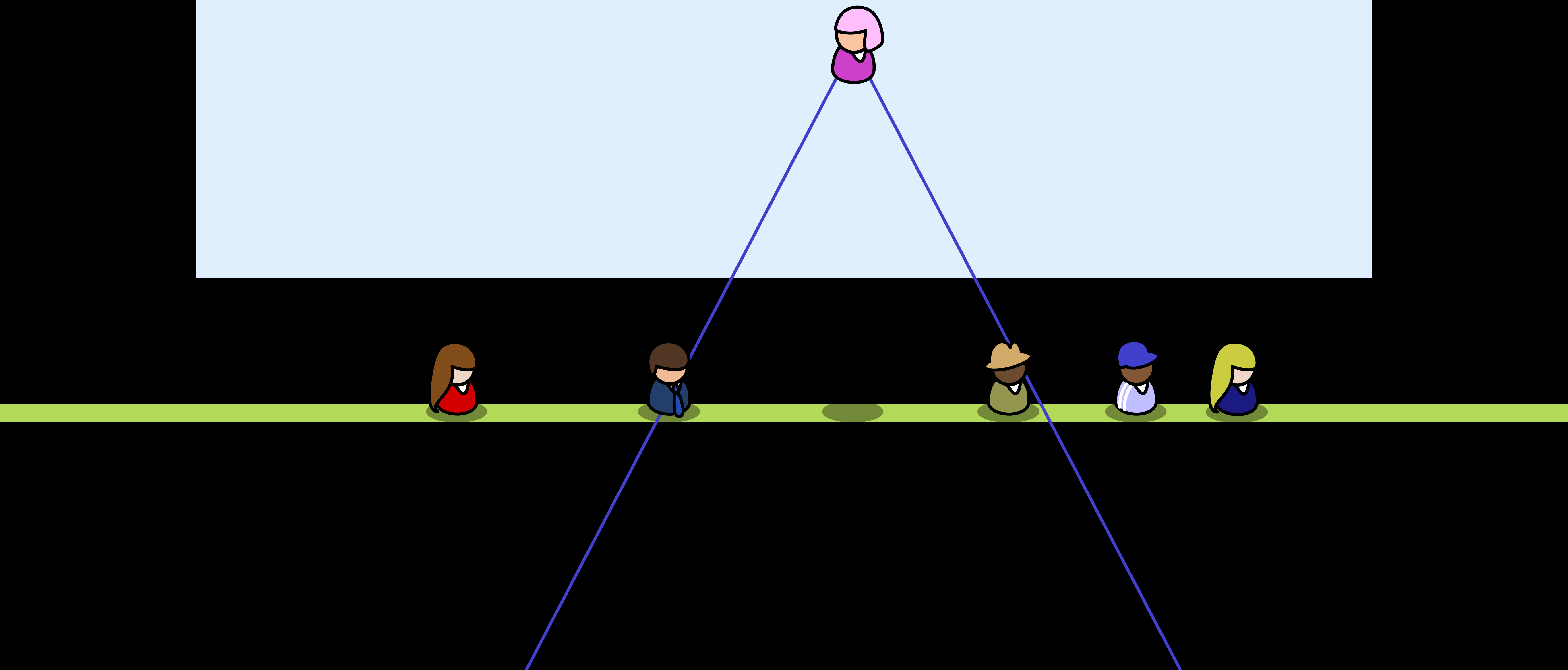
Plan: query each point before it reaches its critical radius



THE STRATEGY

Plan: query each point before it reaches its critical radius

How do we do that?

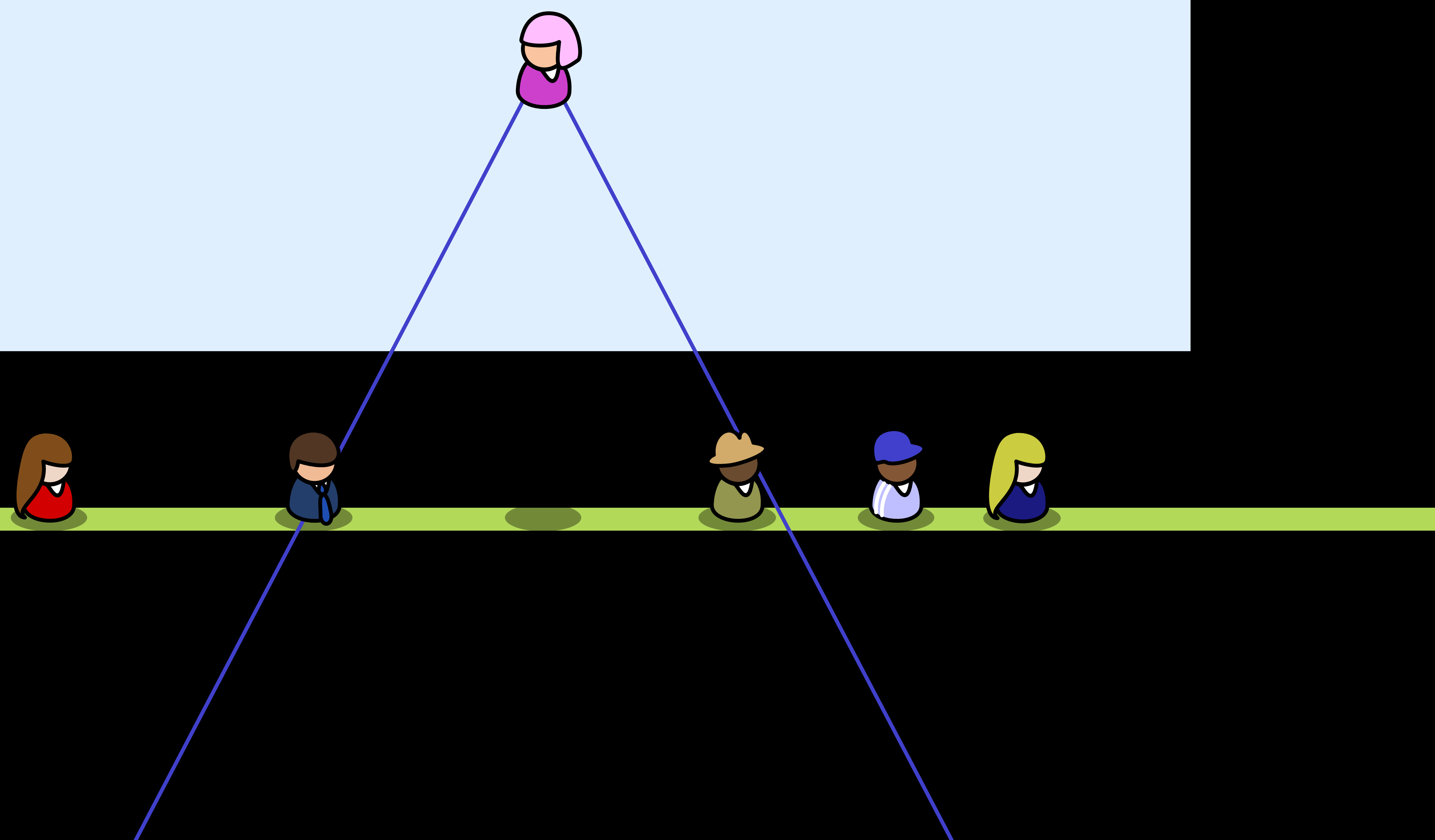


THE STRATEGY

Plan: query each point before it reaches its critical radius

How do we do that?

If $\sum \frac{1}{r_i} > 1$, this is impossible...



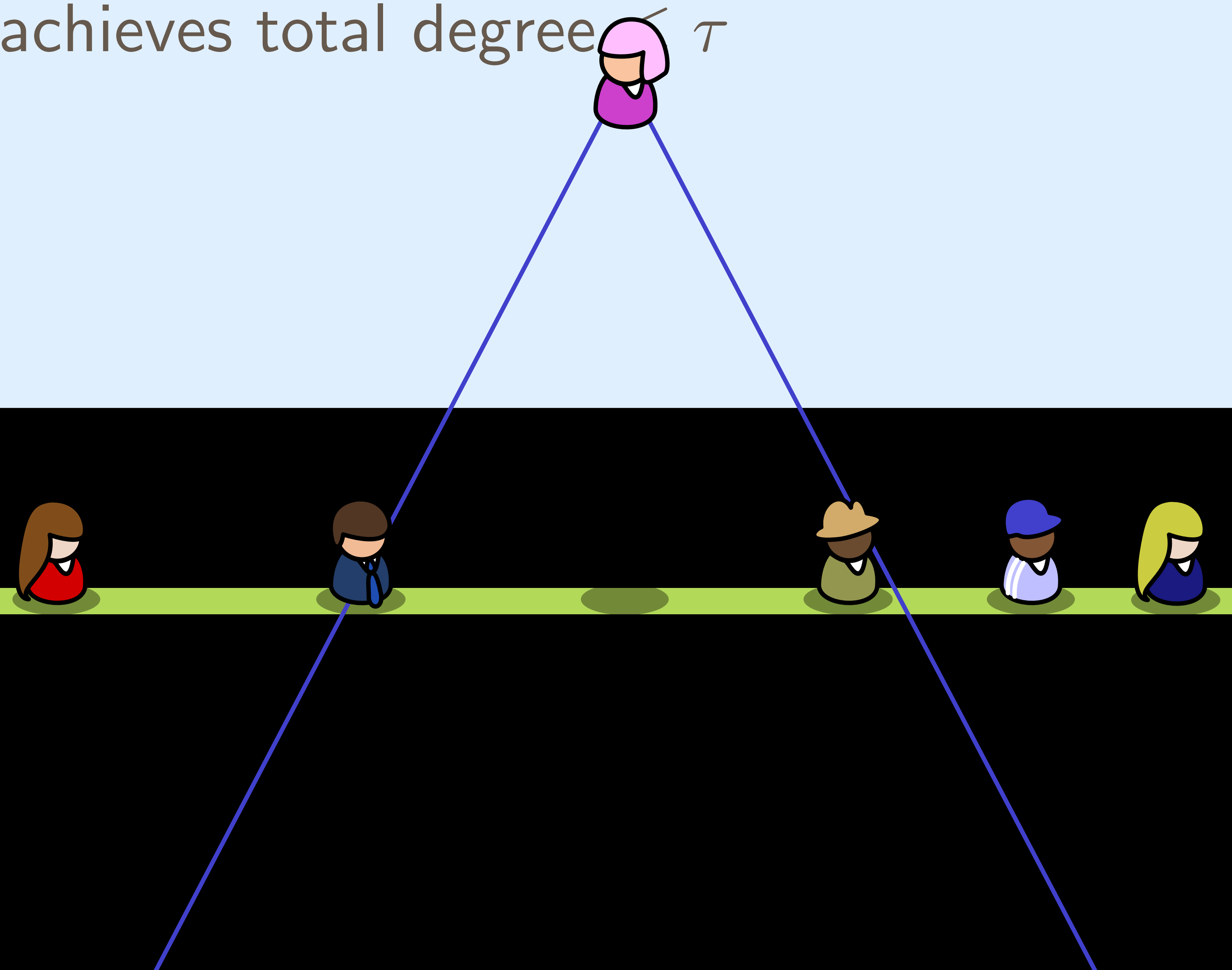
THE STRATEGY

Plan: query each point before it reaches its critical radius

How do we do that?

If $\sum \frac{1}{r_i} > 1$, this is impossible…

… and then there is *no* strategy that achieves total degree $\leq \tau$



THE STRATEGY

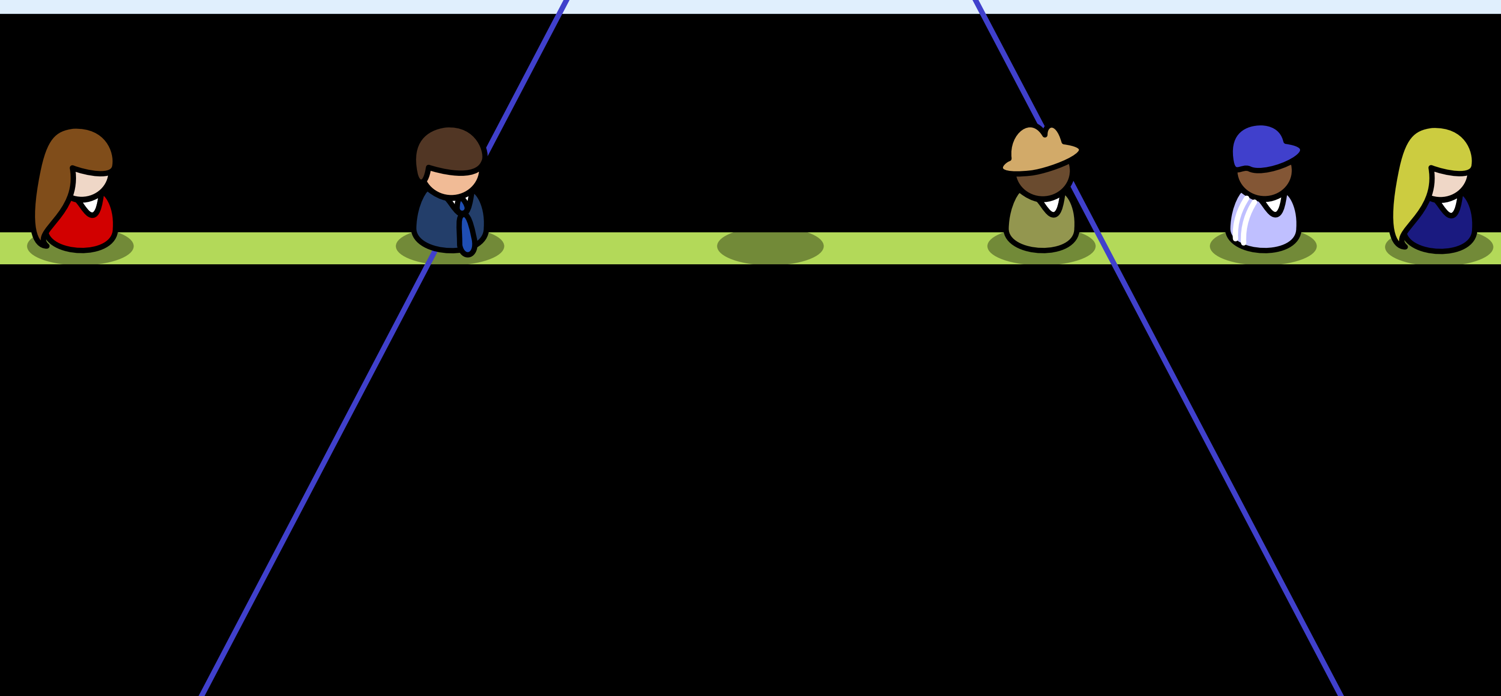
Plan: query each point before it reaches its critical radius

How do we do that?

If $\sum \frac{1}{r_i} > 1$, this is impossible...

... and then there is *no* strategy that achieves total degree $\leq \tau$

Otherwise, solve scheduling problem by grouping points in a quadtree



THE STRATEGY

Plan: query each point before it reaches its critical radius

How do we do that?

- If $\sum \frac{1}{r_i} > 1$, this is impossible...
- ... and then there is *no* strategy that achieves total degree $\leq \tau$
- Otherwise, solve scheduling problem by grouping points in a quadtree

THE STRATEGY

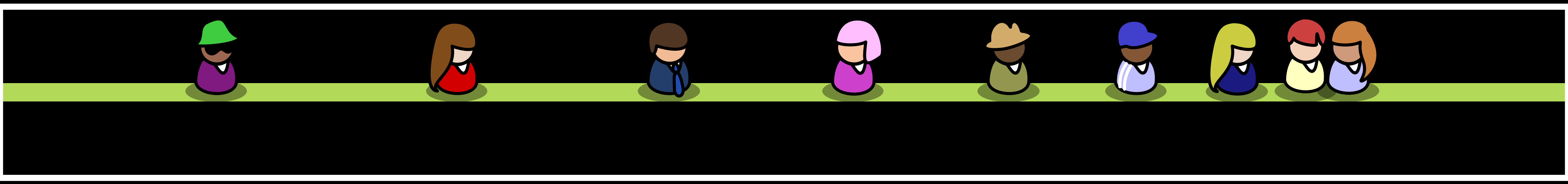
Plan: query each point before it reaches its critical radius

How do we do that?

If $\sum \frac{1}{r_i} > 1$, this is impossible...

... and then there is *no* strategy that achieves total degree $\leq \tau$

Otherwise, solve scheduling problem by grouping points in a quadtree



THE STRATEGY

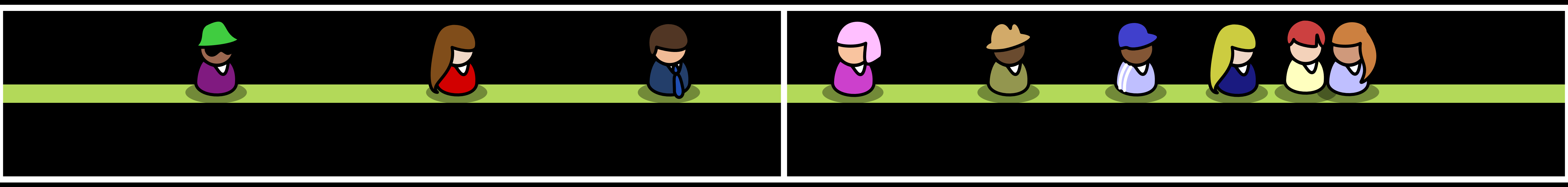
Plan: query each point before it reaches its critical radius

How do we do that?

If $\sum \frac{1}{r_i} > 1$, this is impossible...

... and then there is *no* strategy that achieves total degree $\leq \tau$

Otherwise, solve scheduling problem by grouping points in a quadtree



THE STRATEGY

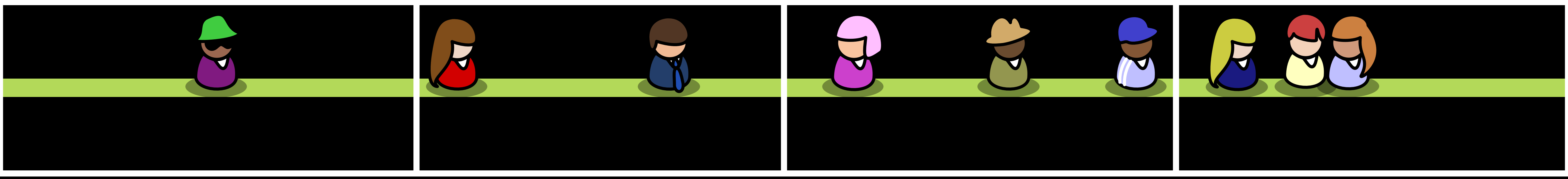
Plan: query each point before it reaches its critical radius

How do we do that?

If $\sum \frac{1}{r_i} > 1$, this is impossible...

... and then there is *no* strategy that achieves total degree $\leq \tau$

Otherwise, solve scheduling problem by grouping points in a quadtree



THE STRATEGY

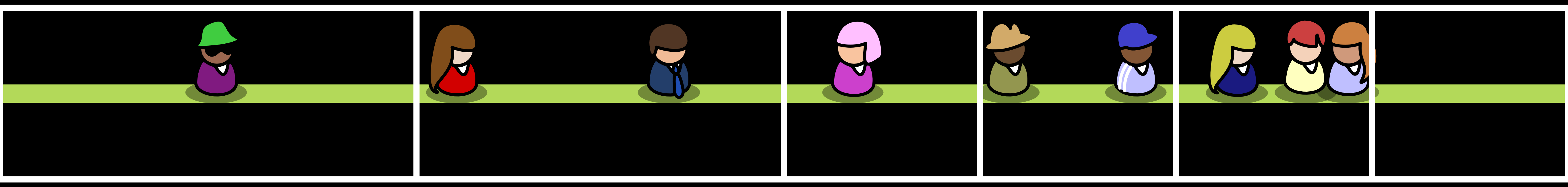
Plan: query each point before it reaches its critical radius

How do we do that?

If $\sum \frac{1}{r_i} > 1$, this is impossible...

... and then there is *no* strategy that achieves total degree $\leq \tau$

Otherwise, solve scheduling problem by grouping points in a quadtree

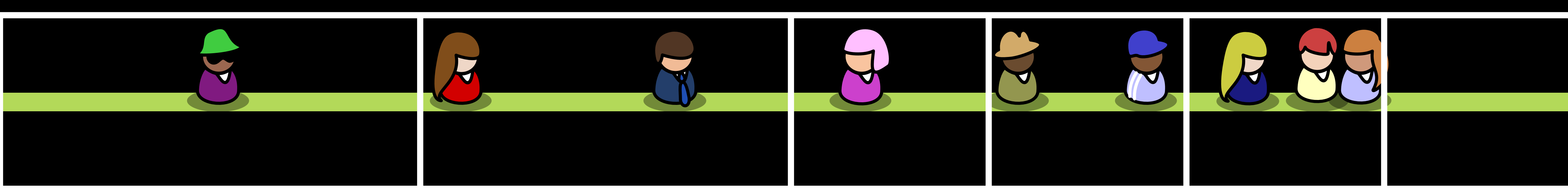


THE STRATEGY

Plan: query each point before it reaches its critical radius

How do we do that?

- If $\sum \frac{1}{r_i} > 1$, this is impossible...
- ... and then there is *no* strategy that achieves total degree $\leq \tau$
- Otherwise, solve scheduling problem by grouping points in a quadtree
- Points in level ℓ get query frequency $\frac{1}{2^\ell}$



OPEN PROBLEM

OPEN PROBLEM

How do we make this *dynamic*?

OPEN PROBLEM

How do we make this *dynamic*?

Problem: critical value τ changes

OPEN PROBLEM

How do we make this *dynamic*?

Problem: critical value τ changes

OPEN PROBLEM

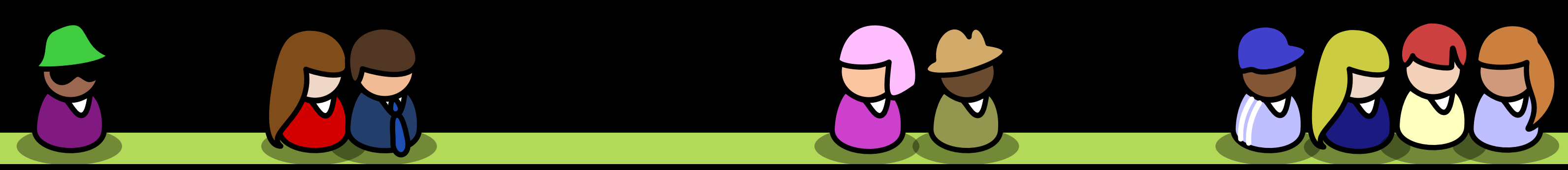
How do we make this *dynamic*?

Problem: critical value τ changes

OPEN PROBLEM

How do we make this *dynamic*?

Problem: critical value τ changes



OPEN PROBLEM

How do we make this *dynamic*?

Problem: critical value τ changes

OPEN PROBLEM

How do we make this *dynamic*?

Problem: critical value τ changes

CONJECTURE

There is a query strategy such that at *every* time t, the critical degree τ *at time* t is maintained during the time interval $[t, t + c\tau]$.

OPEN PROBLEM

How do we make this *dynamic*?

Problem: critical value τ changes

CONJECTURE

There is a query strategy such that at *every* time t, the critical degree τ *at time* t is maintained during the time interval $[t, t + c\tau]$.

THANK YOU!