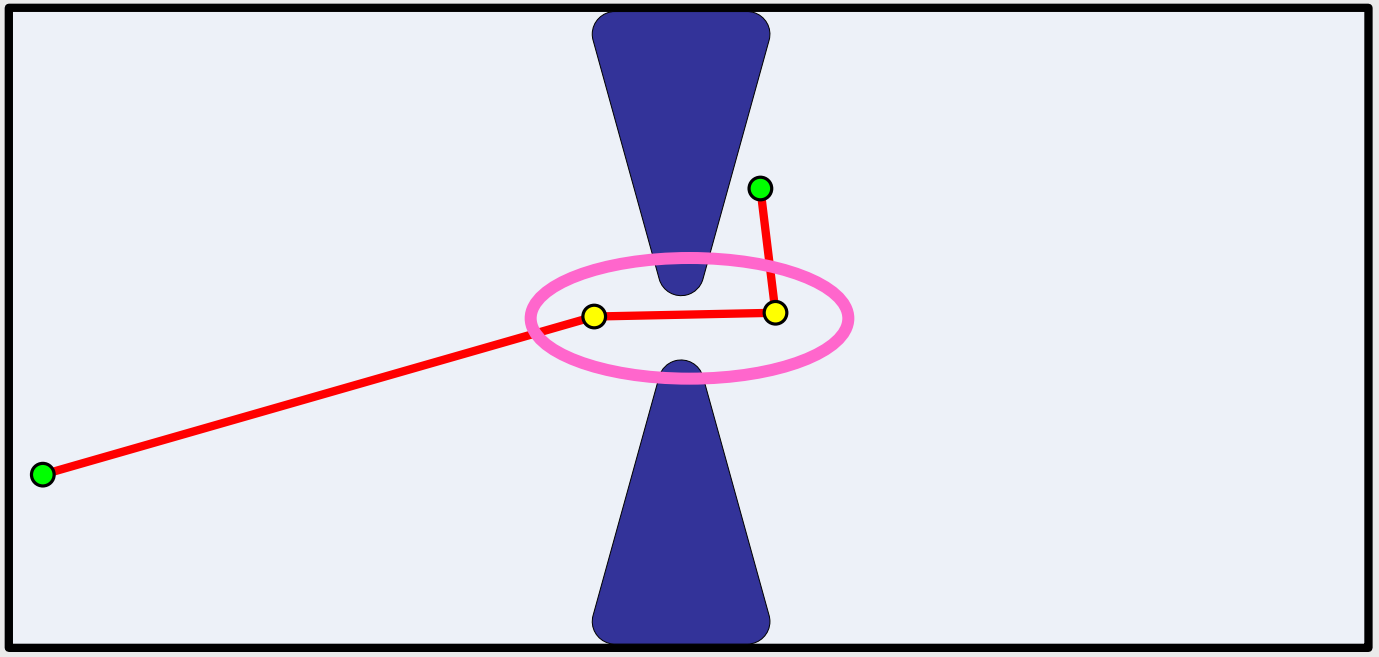
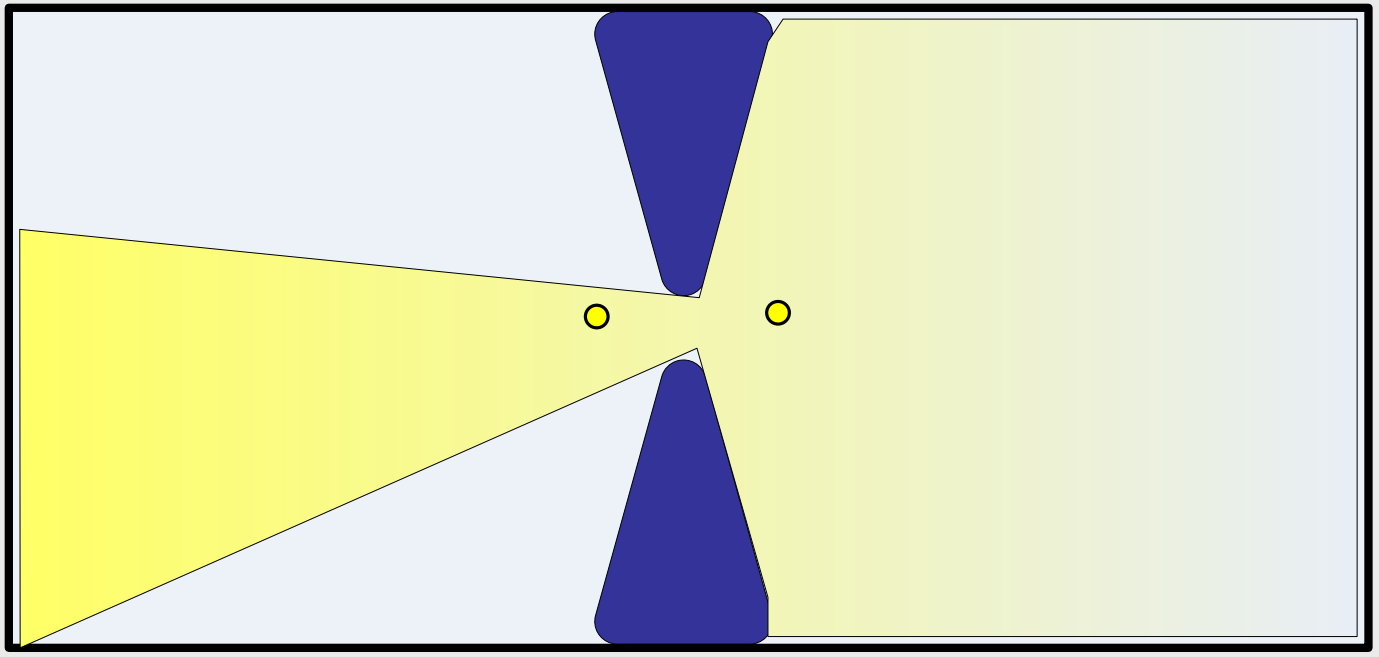
Visibility-PRM: the idea behind



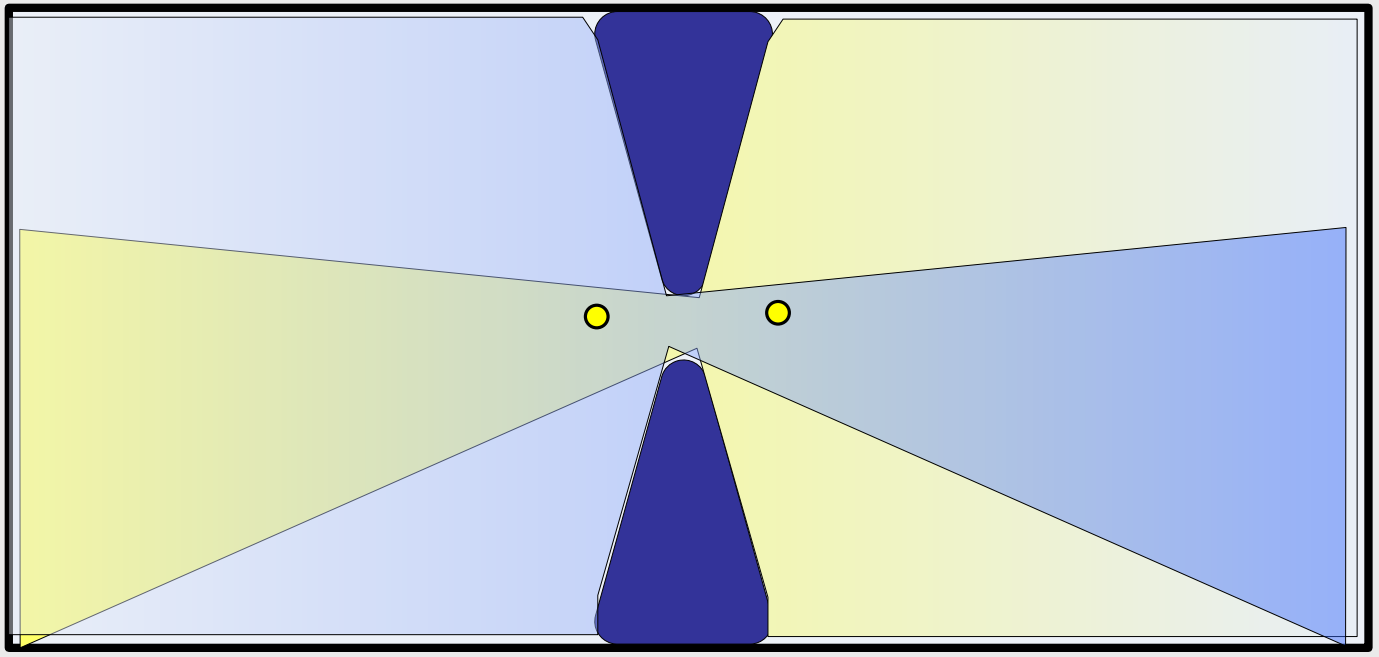
- This minimal „**roadmap**“ is valid for every start & goal position
- Why?

Visibility-PRM: the idea behind



- This minimal „**roadmap**“ is valid for every start & goal position
- Why? All possible points are visible from the two of the roadmap

Visibility-PRM: the idea behind



- This minimal „**roadmap**“ is valid for every start & goal position
- Why? All possible points are **visible** from the two of the roadmap

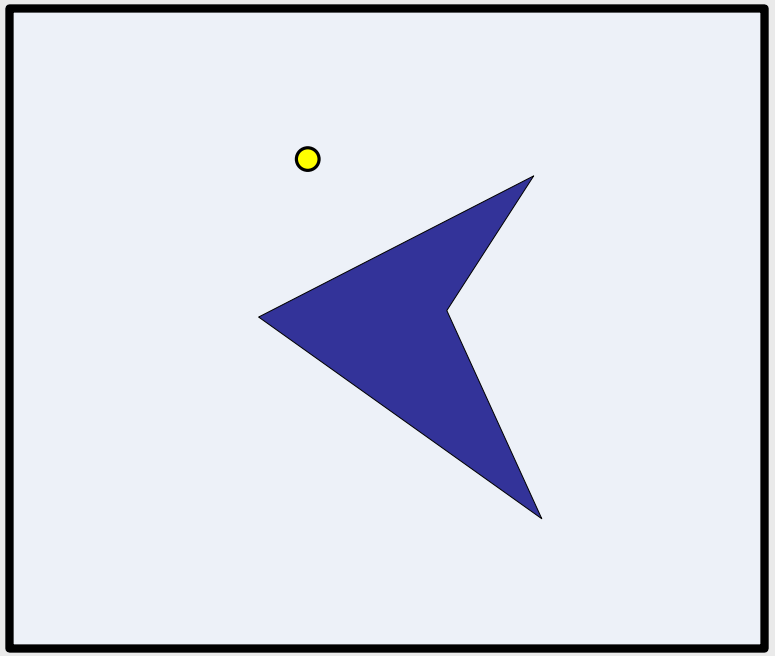
Some Terms

- Visibility domain, Guard and Connection:
  - Let **L**(q,q´) be **any local method** that computes a path between to configurations q and q´.
  - Visibility domain: $\mathbf{Vis_L}(q)$ = { all q´ in CS_{free} such that **L**(q,q´) in CS_{free}}.
  - q is then called **guard** of $\mathbf{Vis_L}(q)$.
  - a configuration is called **connection** if it lies within the visibility domain of two or more guards.

Collision free

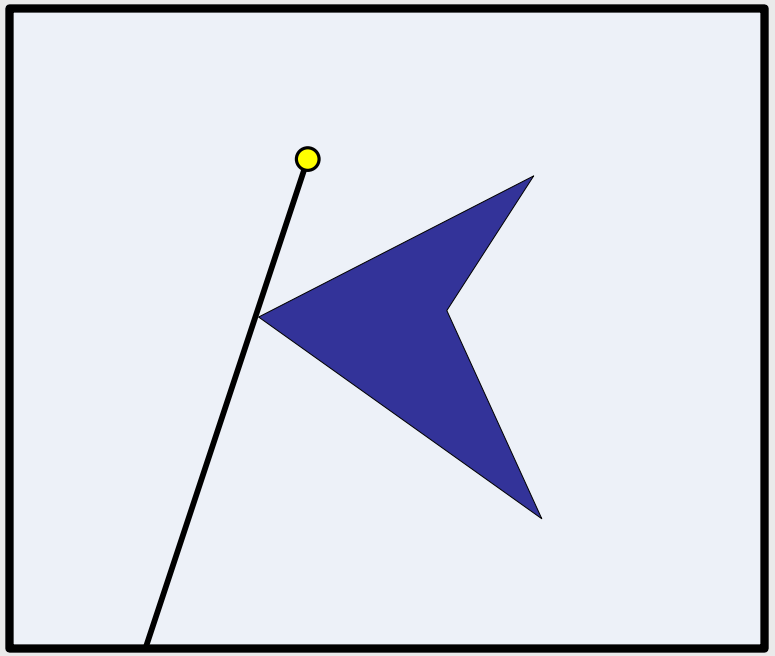
Some Terms

- Visibility domain, Guard and Connection:
  - Let **L**(q,q´) be **any local method** that computes a path between to configurations q and q´.
  - Visibility domain: $\mathbf{Vis_L}(q)$ = { all q´ in CS_{free} such that **L**(q,q´) in CS_{free}}.
  - q is then called **guard** of $\mathbf{Vis_L}(q)$.
  - a configuration is called **connection** if it lies within the visibility domain of two or more guards.
  - Example:
    (local method = straight line tester):



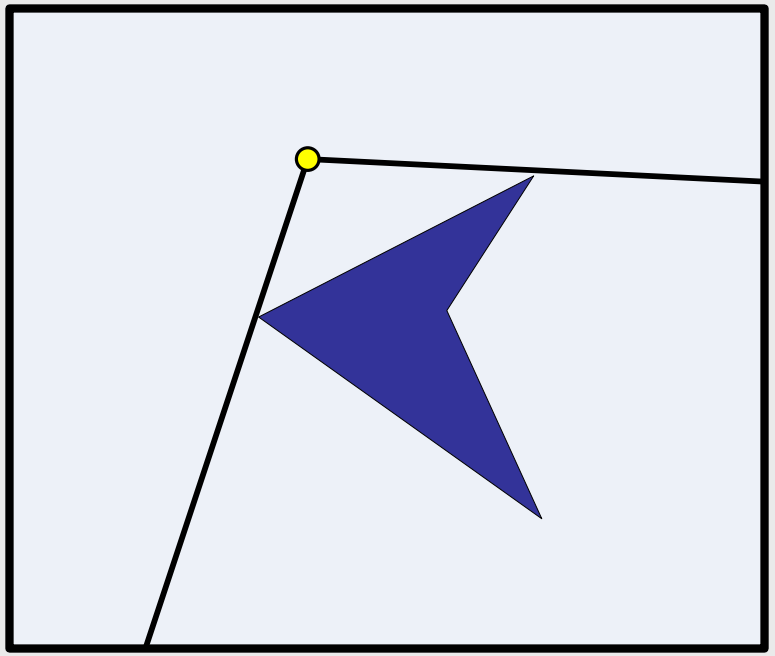
Some Terms

- Visibility domain, Guard and Connection:
  - Let **L**(q,q´) be **any local method** that computes a path between to configurations q and q´.
  - Visibility domain: $\mathbf{Vis_L}(q)$ = { all q´ in CS_{free} such that **L**(q,q´) in CS_{free}}.
  - q is then called **guard** of $\mathbf{Vis_L}(q)$.
  - a configuration is called **connection** if it lies within the visibility domain of two or more guards.
  - Example:
    (local method = straight line tester):



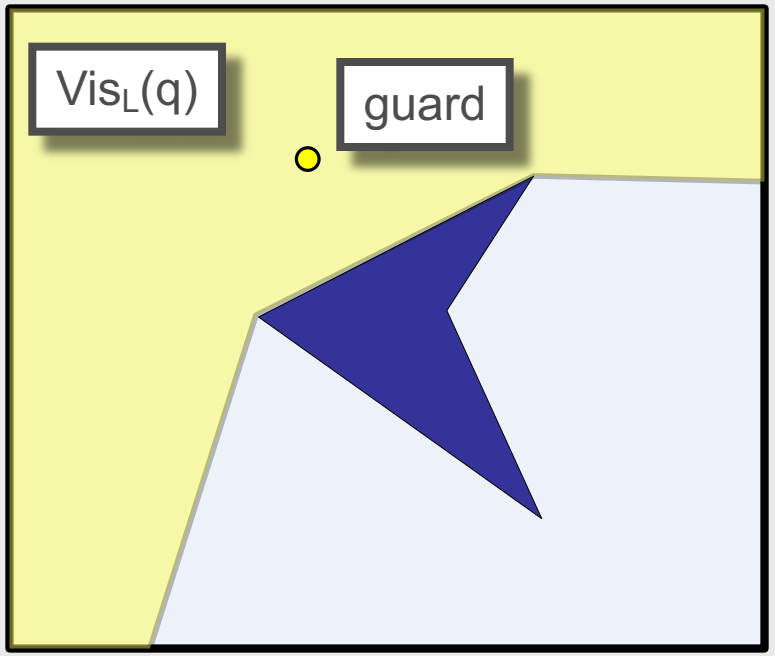
Some Terms

- Visibility domain, Guard and Connection:
  - Let **L**(q,q´) be **any local method** that computes a path between to configurations q and q´.
  - Visibility domain: $\mathbf{Vis_L}(q)$ = { all q´ in CS_{free} such that **L**(q,q´) in CS_{free}}.
  - q is then called **guard** of $\mathbf{Vis_L}(q)$.
  - a configuration is called **connection** if it lies within the visibility domain of two or more guards.
  - Example:
    (local method = straight line tester):



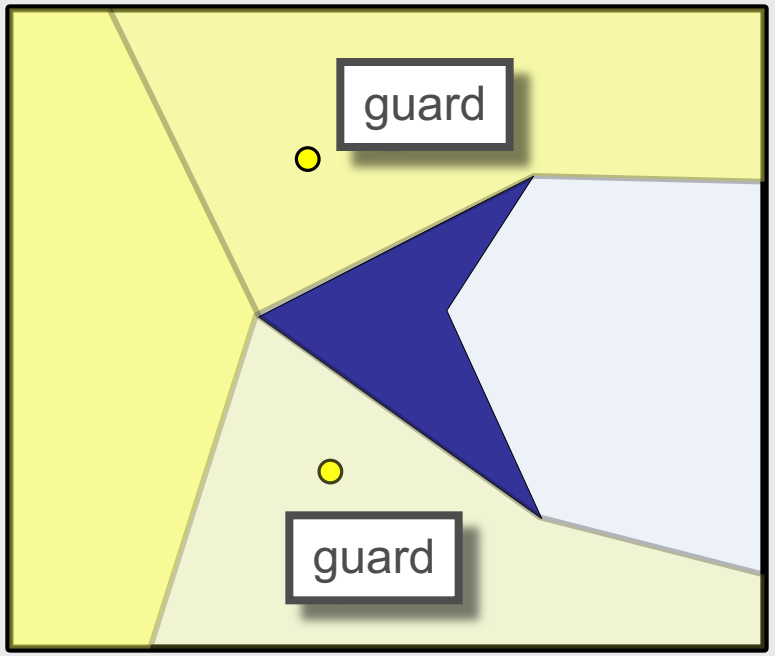
Some Terms

- Visibility domain, Guard and Connection:
  - Let **L**(q,q´) be **any local method** that computes a path between to configurations q and q´.
  - Visibility domain: $\mathbf{Vis_L}(q)$ = { all q´ in CS_{free} such that **L**(q,q´) in CS_{free}}.
  - q is then called **guard** of $\mathbf{Vis_L}(q)$.
  - a configuration is called **connection** if it lies within the visibility domain of two or more guards.
  - Example:
    (local method = straight line tester):



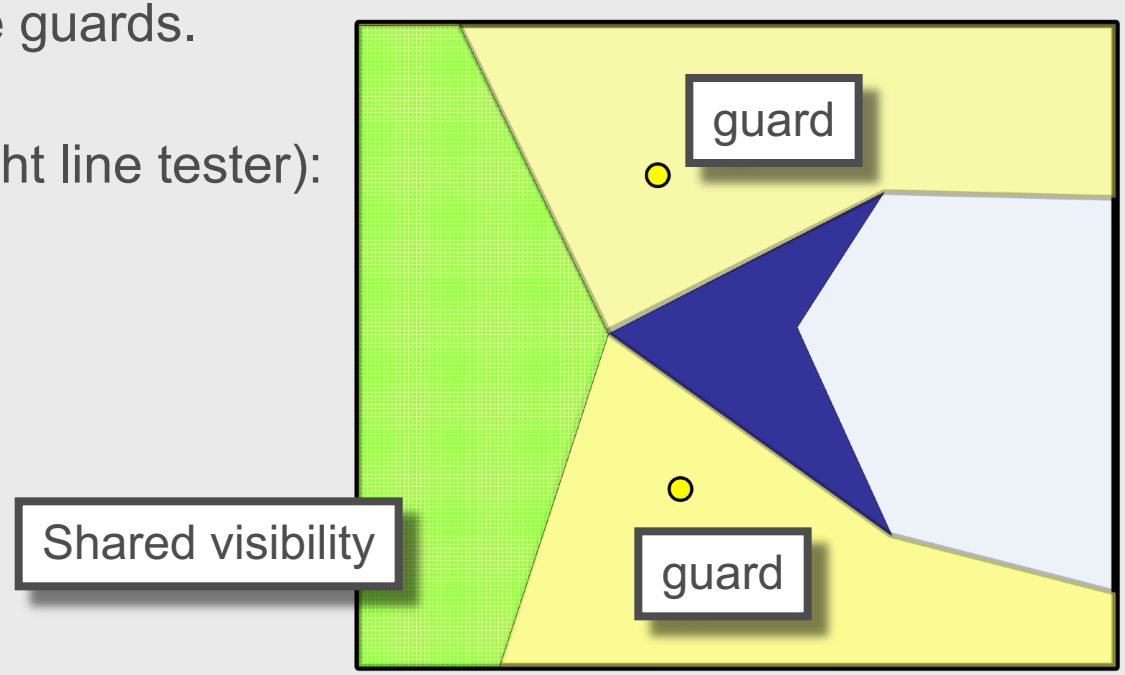
Some Terms

- Visibility domain, Guard and Connection:
  - Let **L**(q,q´) be **any local method** that computes a path between to configurations q and q´.
  - Visibility domain: $\mathbf{Vis_L}(q)$ = { all q´ in CS_{free} such that **L**(q,q´) in CS_{free}}.
  - q is then called **guard** of $\mathbf{Vis_L}(q)$.
  - a configuration is called **connection** if it lies within the visibility domain of two or more guards.
  - Example:
    (local method = straight line tester):



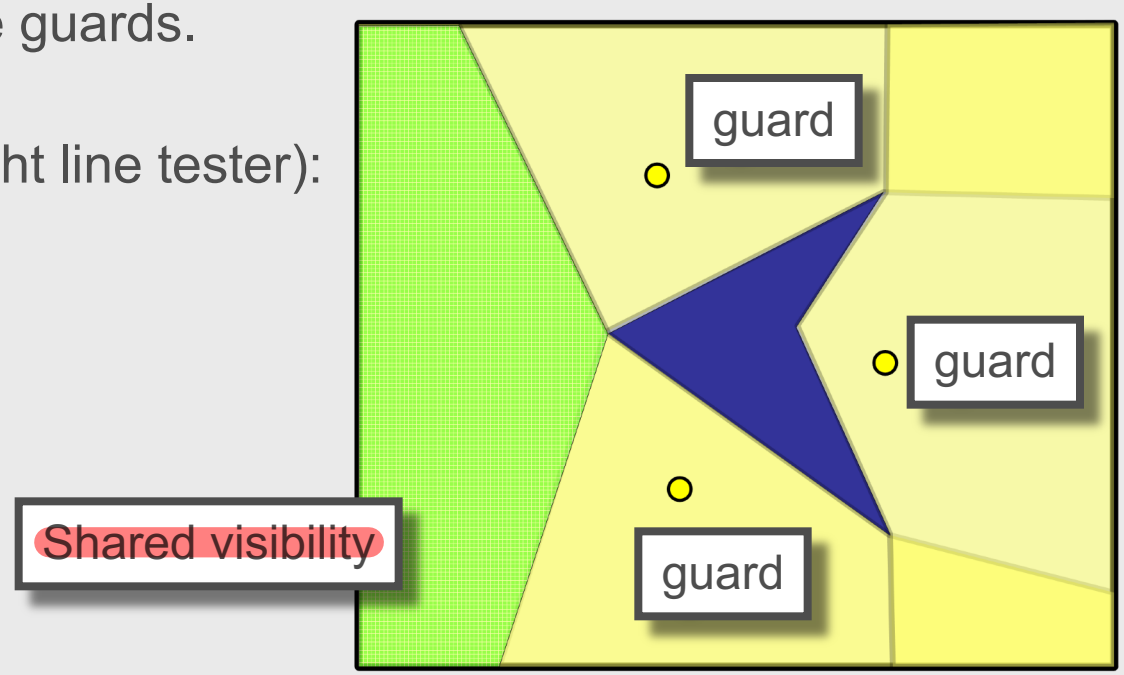
Some Terms

- Visibility domain, Guard and Connection:
  - Let **L**(q,q´) be **any local method** that computes a path between to configurations q and q´.
  - Visibility domain: $\mathbf{Vis_L}(q)$ = { all q´ in CS_{free} such that **L**(q,q´) in CS_{free}}.
  - q is then called **guard** of $\mathbf{Vis_L}(q)$.
  - a configuration is called **connection** if it lies within the visibility domain of two or more guards.
  - Example:
    (local method = straight line tester):



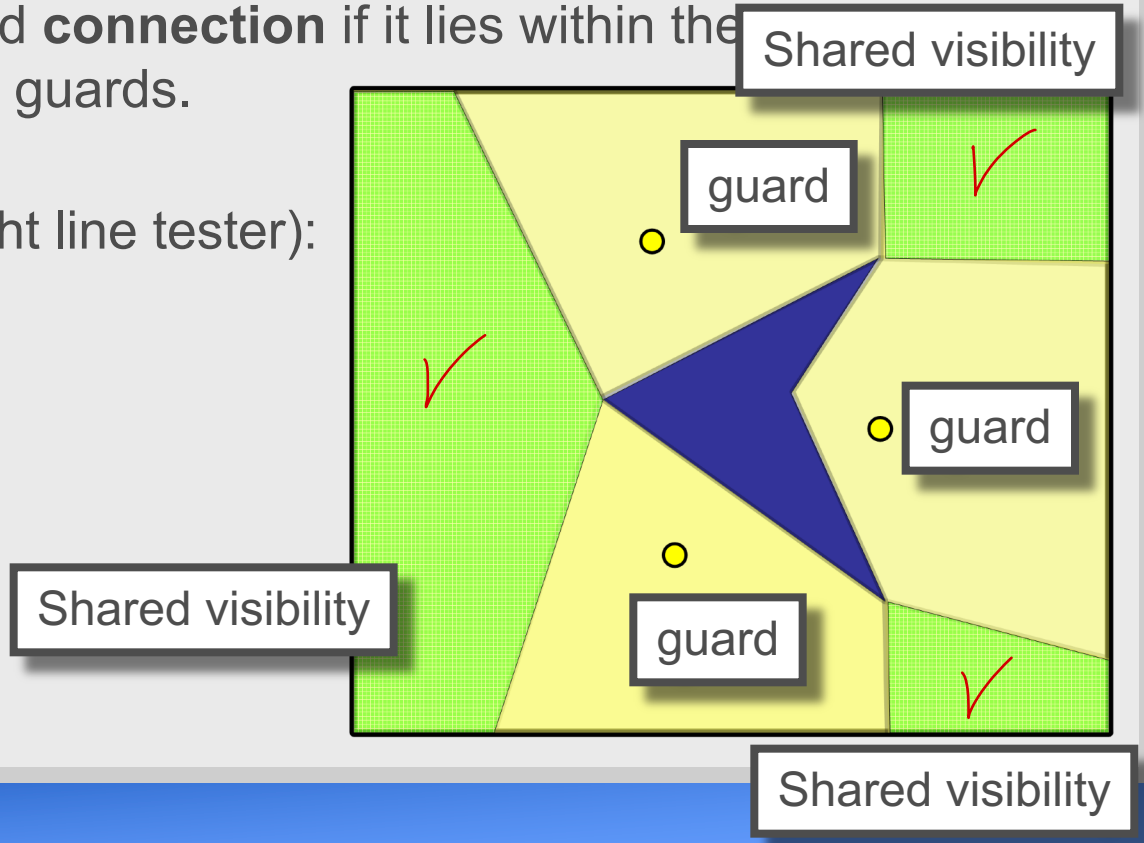
Some Terms

- Visibility domain, Guard and Connection:
  - Let **L**(q,q´) be **any local method** that computes a path between to configurations q and q´.
  - Visibility domain: $\mathbf{Vis_L}(q)$ = { all q´ in CS_{free} such that **L**(q,q´) in CS_{free}}.
  - q is then called **guard** of $\mathbf{Vis_L}(q)$.
  - a configuration is called **connection** if it lies within the visibility domain of two or more guards.
  - Example:
    (local method = straight line tester):



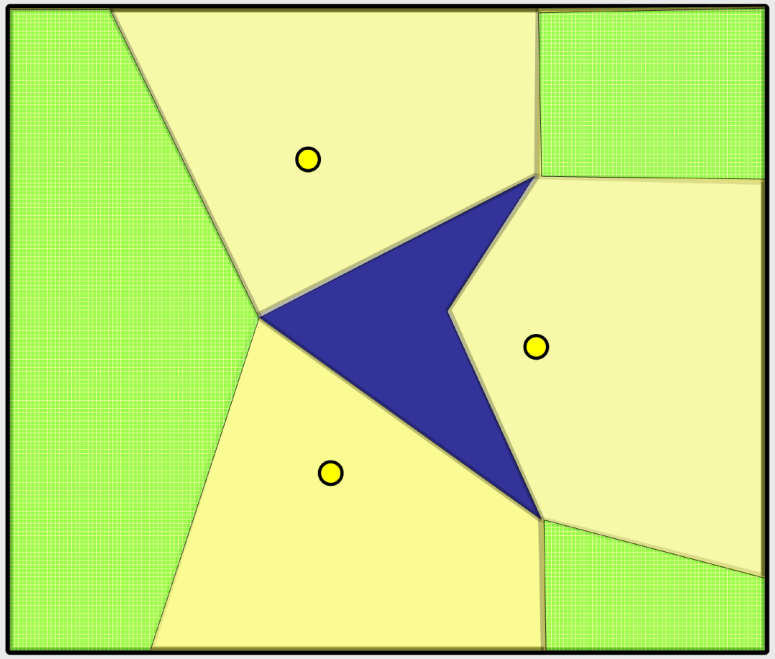
Some Terms

- Visibility domain, Guard and Connection:
  - Let **L**(q,q´) be **any local method** that computes a path between to configurations q and q´.
  - Visibility domain: $\mathbf{Vis_L}(q)$ = { all q´ in CS_{free} such that **L**(q,q´) in CS_{free}}.
  - q is then called **guard** of $\mathbf{Vis_L}(q)$.
  - a configuration is called **connection** if it lies within the domain of two or more guards.
  - Example:
    (local method = straight line tester):



Some Terms

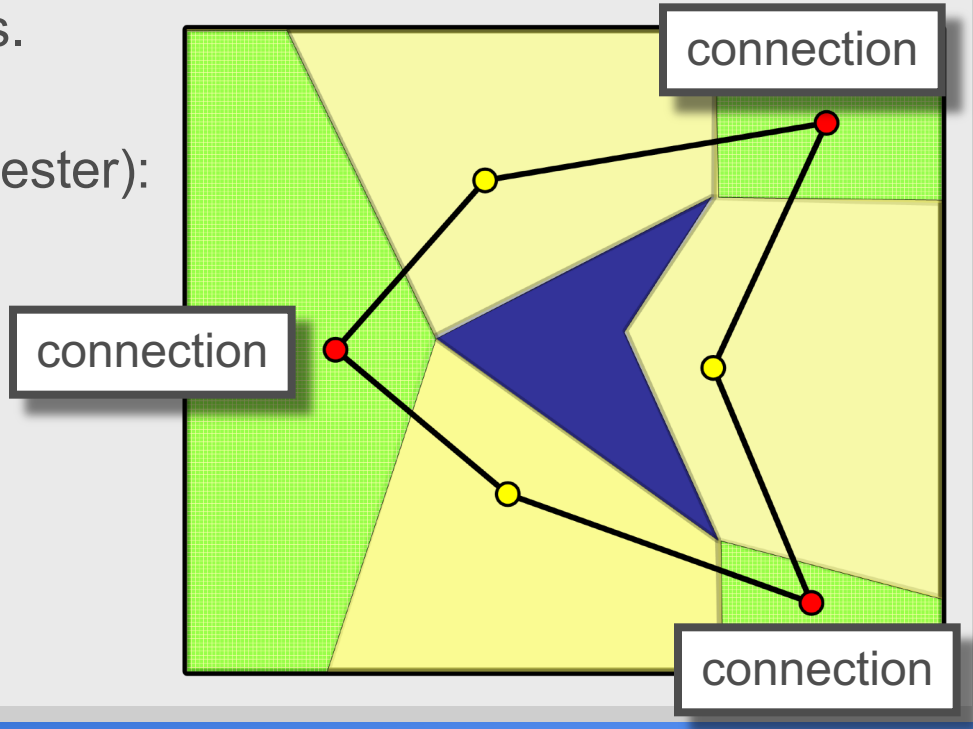
- Visibility domain, Guard and Connection:
  - Let **L**(q,q´) be **any local method** that computes a path between to configurations q and q´.
  - Visibility domain: $\mathbf{Vis_L}(q)$ = { all q´ in CS_{free} such that **L**(q,q´) in CS_{free}}.
  - q is then called **guard** of $\mathbf{Vis_L}(q)$.
  - a configuration is called **connection** if it lies within the visibility domain of two or more guards.
  - Example:
    (local method = straight line tester):



Some Terms

- Visibility domain, Guard and Connection:
  - Let **L**(q,q´) be **any local method** that computes a path between to configurations q and q´.
  - Visibility domain: $\mathbf{Vis_L}(q)$ = { all q´ in CS_{free} such that **L**(q,q´) in CS_{free}}.
  - q is then called **guard** of $\mathbf{Vis_L}(q)$.
  - a configuration is called **connection** if it lies within the visibility domain of two or more guards.
  - Example:
    (local method = straight line tester):

guard ist ein Node, von dem aus ein Viewfield erzeugt wird.



3 nodes are enough to create a complete path

Visibility-PRM: Build up the roadmap

```
Input M = Number of tests to be done

Guard = ∅; Connection = ∅; ntry = 0
while (ntry < M)
    Select a random free configuration q
    g_vis = ∅; G_Vis = ∅
    for all components G_i of Guard do
        found = FALSE
        for all nodes g of G_i do
          if ( q belongs to Vis(g) ) then
            found = TRUE
            if ( !g_vis ) then
              g_vis = g; G_vis = G_i
            else /* q is a connection node */
              add q to Connection
              Create edges (q,g) and (q, g_vis)
              Merge components G_vis and G_i
        until found = TRUE
    if ( !g_vis ) then /* q is a guard node */
        add {q} to Guard; ntry = 0
    else ntry = ntry +1
end
```