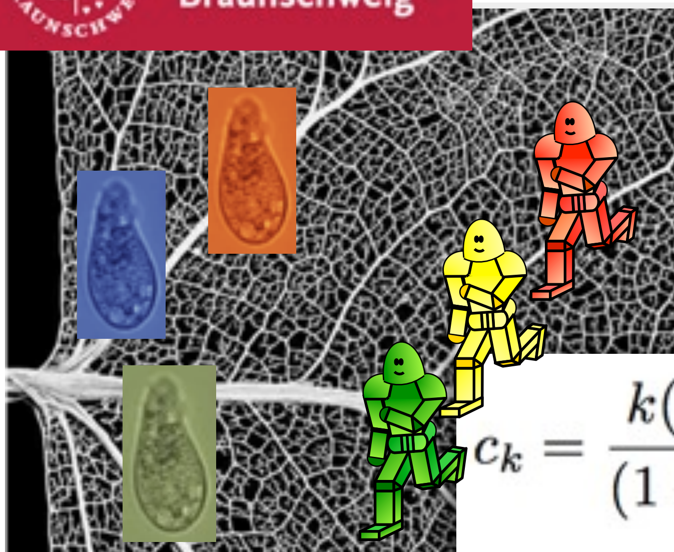
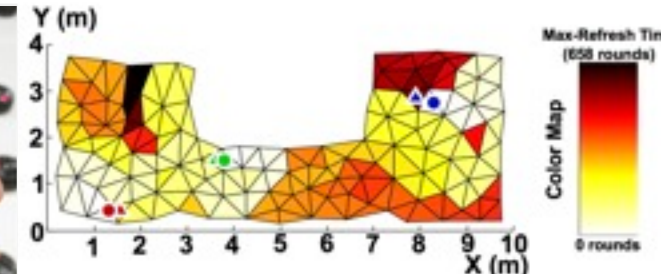
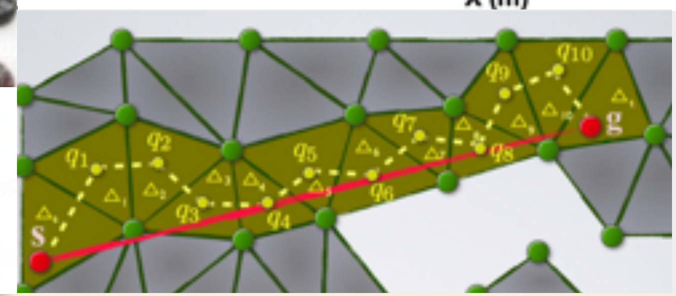
Technische Universität Braunschweig



$$c_k = \frac{k(1+\frac{z_k}{k})-(k-1)}{(1+\frac{z_k}{k})-\frac{1}{(1+\frac{z_k}{k})^{k-1}}}$$





Online Navigation for Robots

Pravesh Agrawal, Aaron Becker, Erik D. Demaine

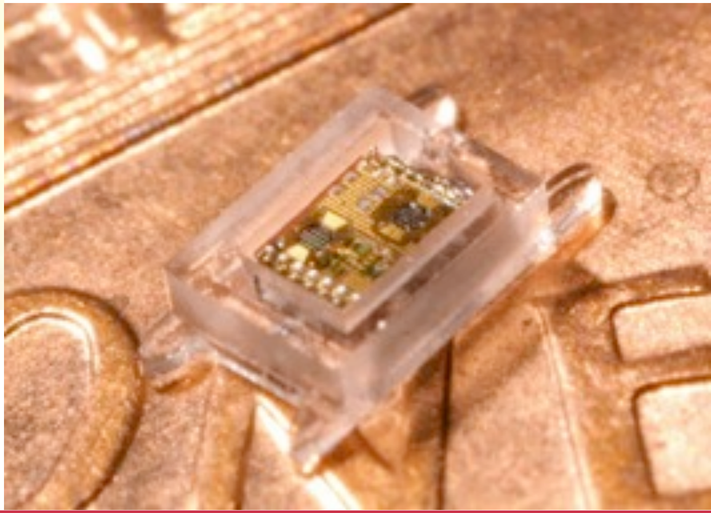
Sándor P. Fekete

Golnaz Habibi, Rolf Klein, Alexander Kröller, Andreas Nüchter

Seoung Kyou Lee, James McLurkin, Christiane Schmidt

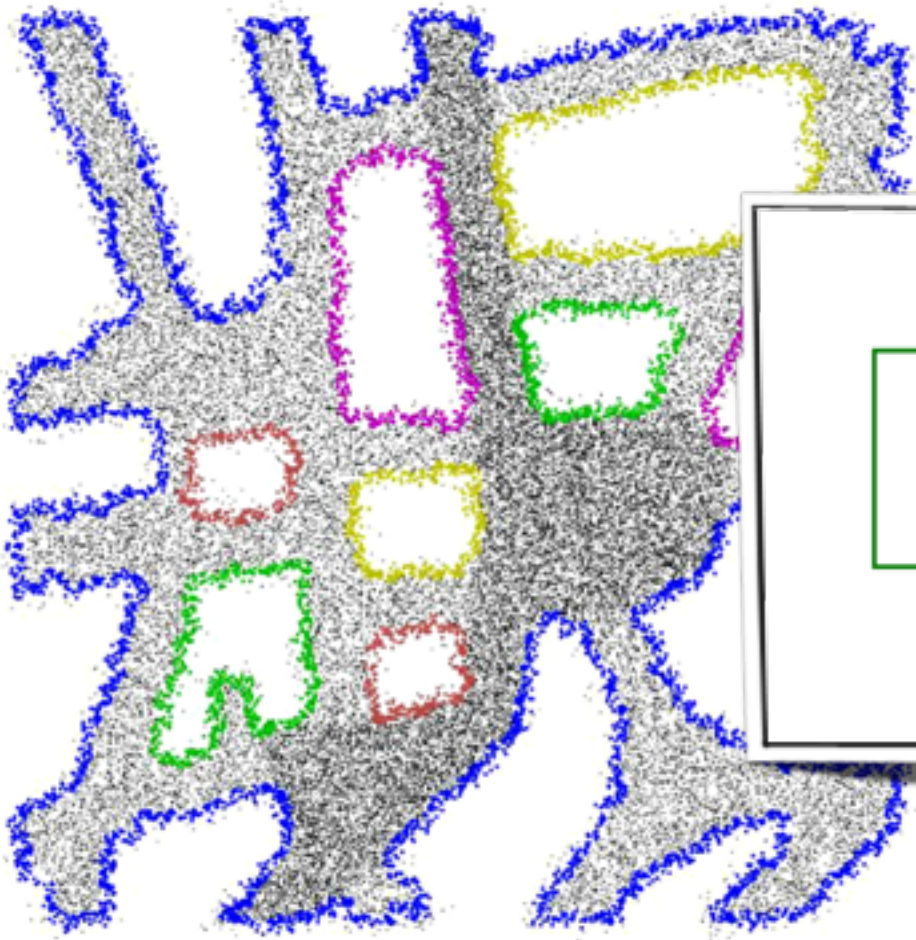
Preface: Processors and Mobile Objects

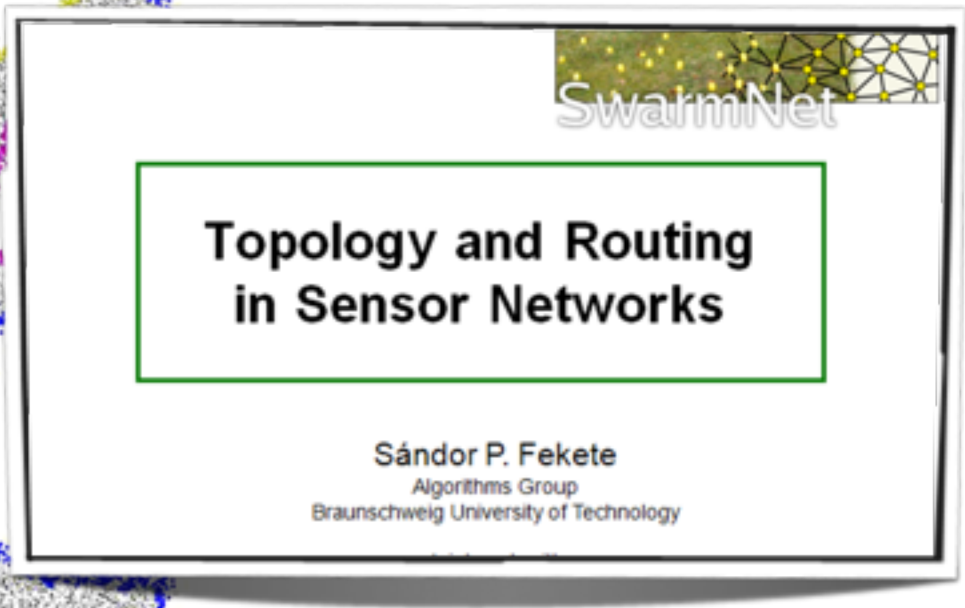
Processors



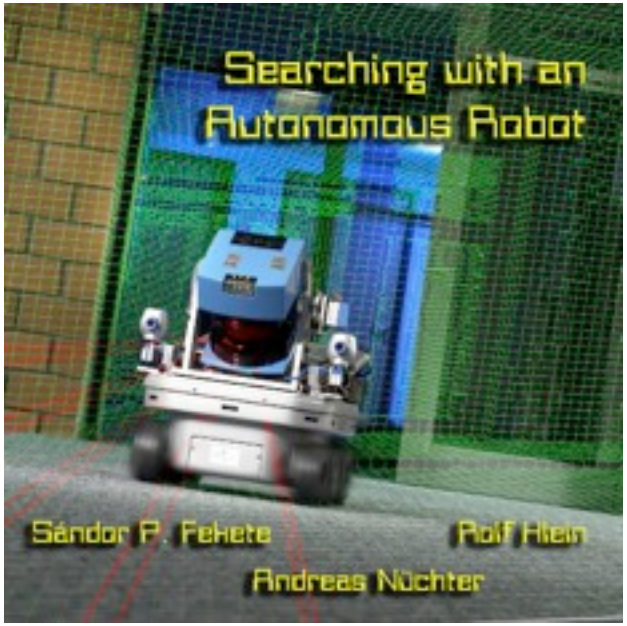
“Smart Dust”

“Smart Dust”



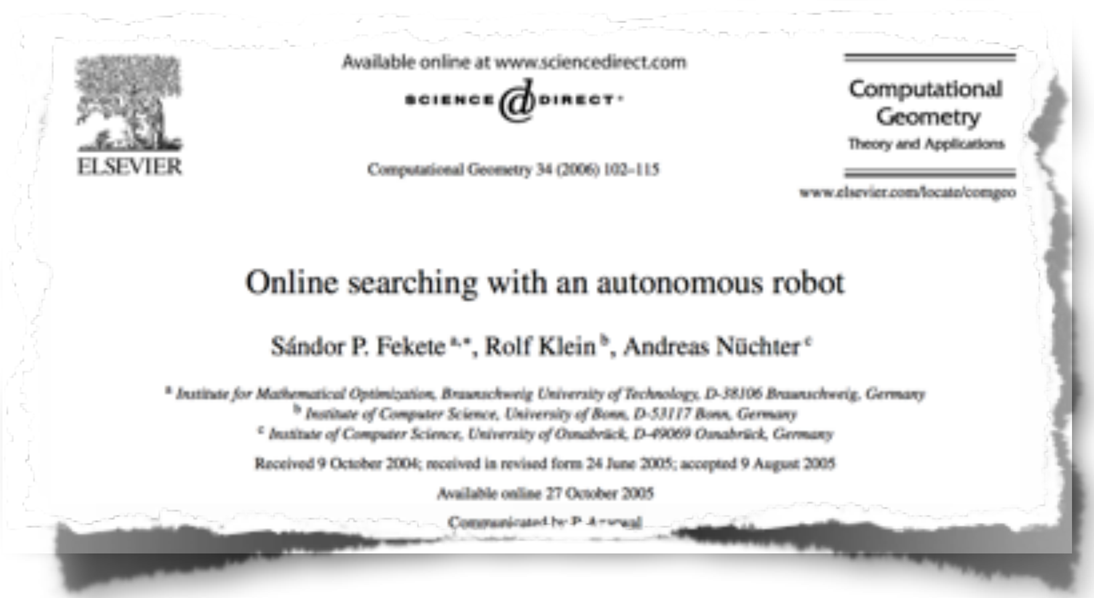

SwarmNet
Topology and Routing in Sensor Networks
Sándor P. Fekete
Algorithms Group
Braunschweig University of Technology

Mobile Objects and Robots



Part 1: One Robot

Part 1.1:
Looking around a corner


Available online at www.sciencedirect.com
SCIENCE DIRECT
Computational Geometry 34 (2006) 102–115
Computational Geometry Theory and Applications
www.elsevier.com/locate/comgeo

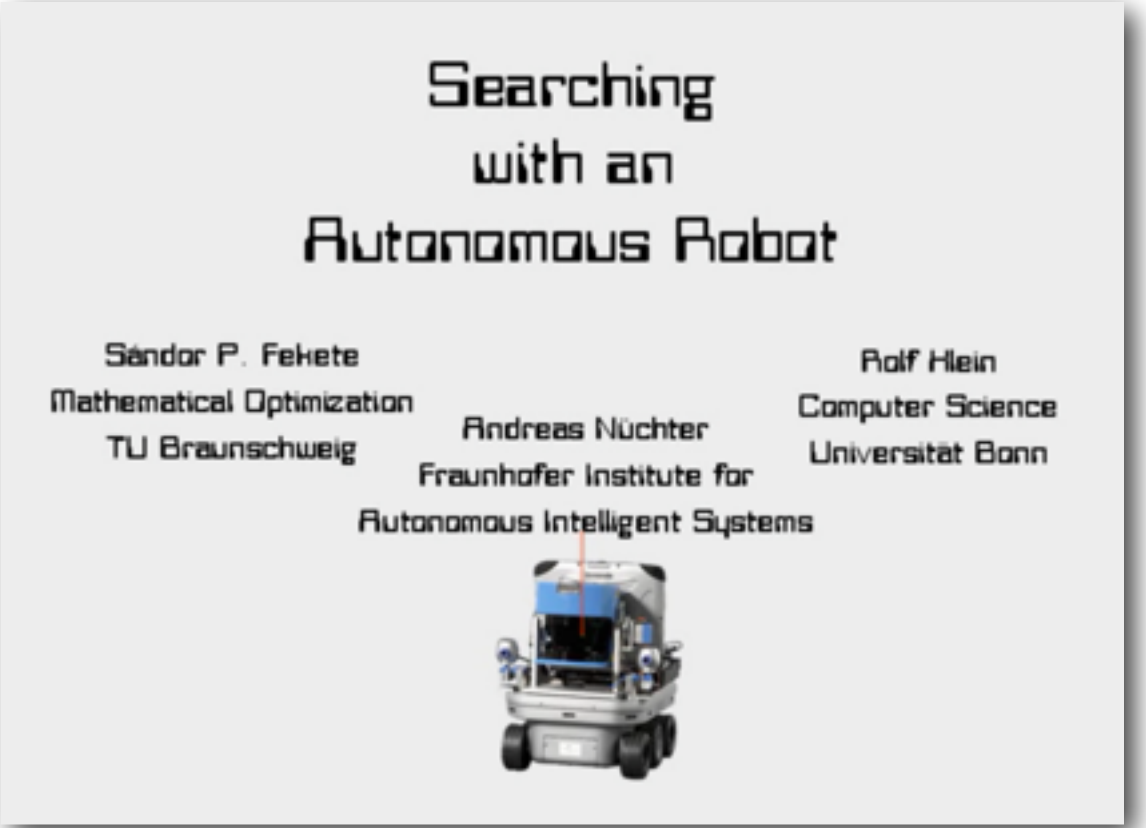
Online searching with an autonomous robot

Sándor P. Fekete [a,*], Rolf Klein [b], Andreas Nüchter [c]

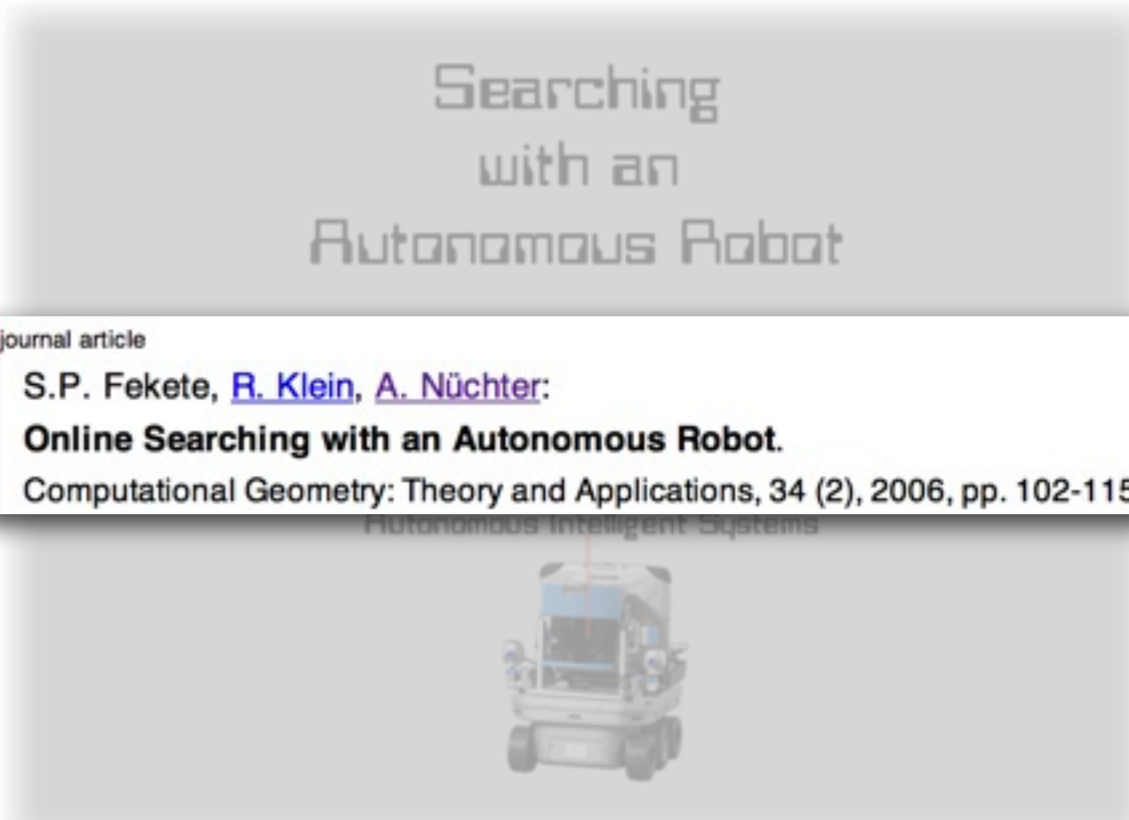
[a] *Institute for Mathematical Optimization, Braunschweig University of Technology, D-38106 Braunschweig, Germany*
[b] *Institute of Computer Science, University of Bonn, D-53117 Bonn, Germany*
[c] *Institute of Computer Science, University of Osnabrück, D-49069 Osnabrück, Germany*

Received 9 October 2004; received in revised form 24 June 2005; accepted 9 August 2005
Available online 27 October 2005
Communicated by P. A[illegible]

Video!


Searching
with an
Autonomous Robot
Sándor P. Fekete
Mathematical Optimization
TU Braunschweig
Andreas Nüchter
Fraunhofer Institute for
Autonomous Intelligent Systems
Rolf Klein
Computer Science
Universität Bonn

Video!



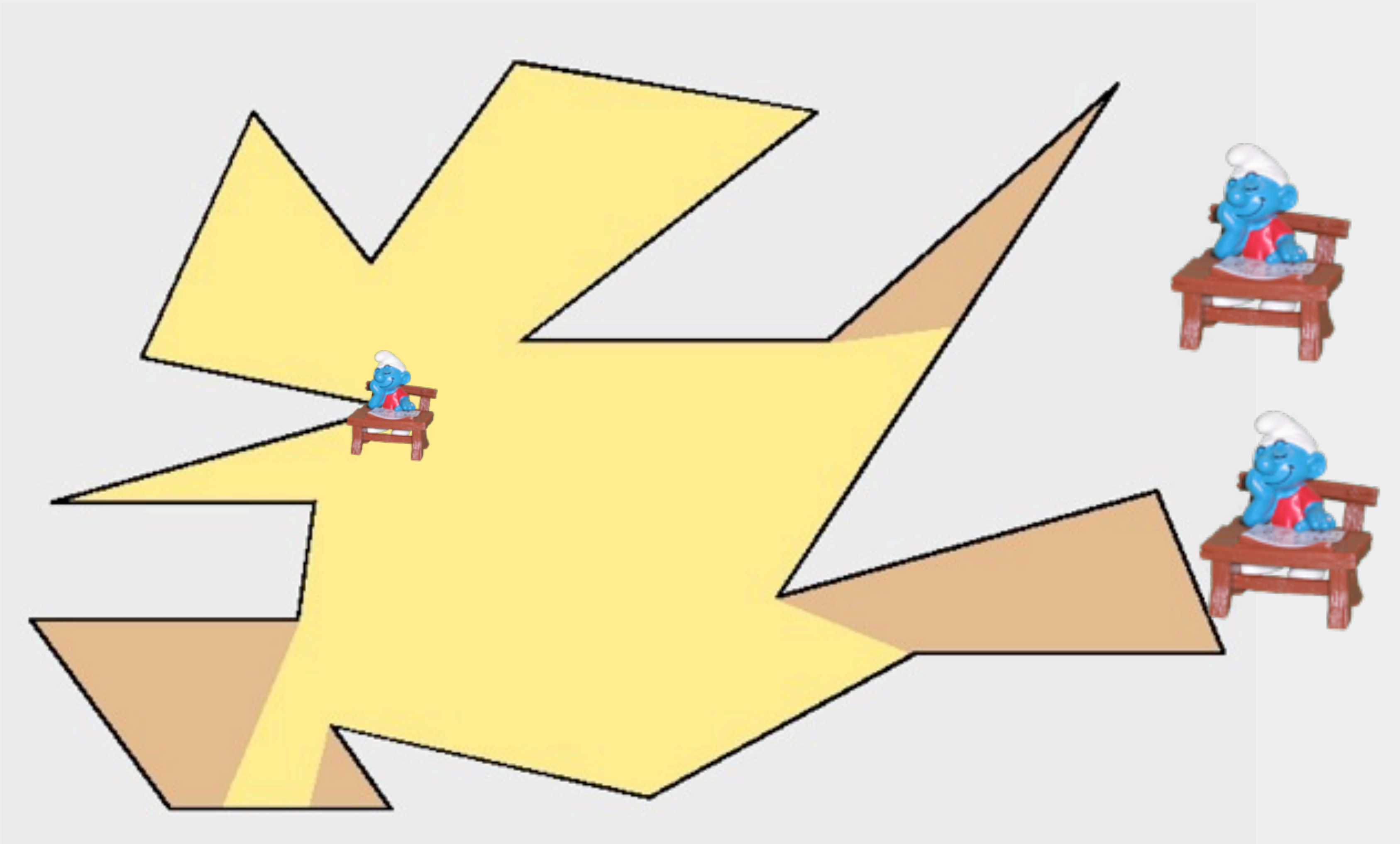
journal article

S.P. Fekete, R. Klein, A. Nüchter:

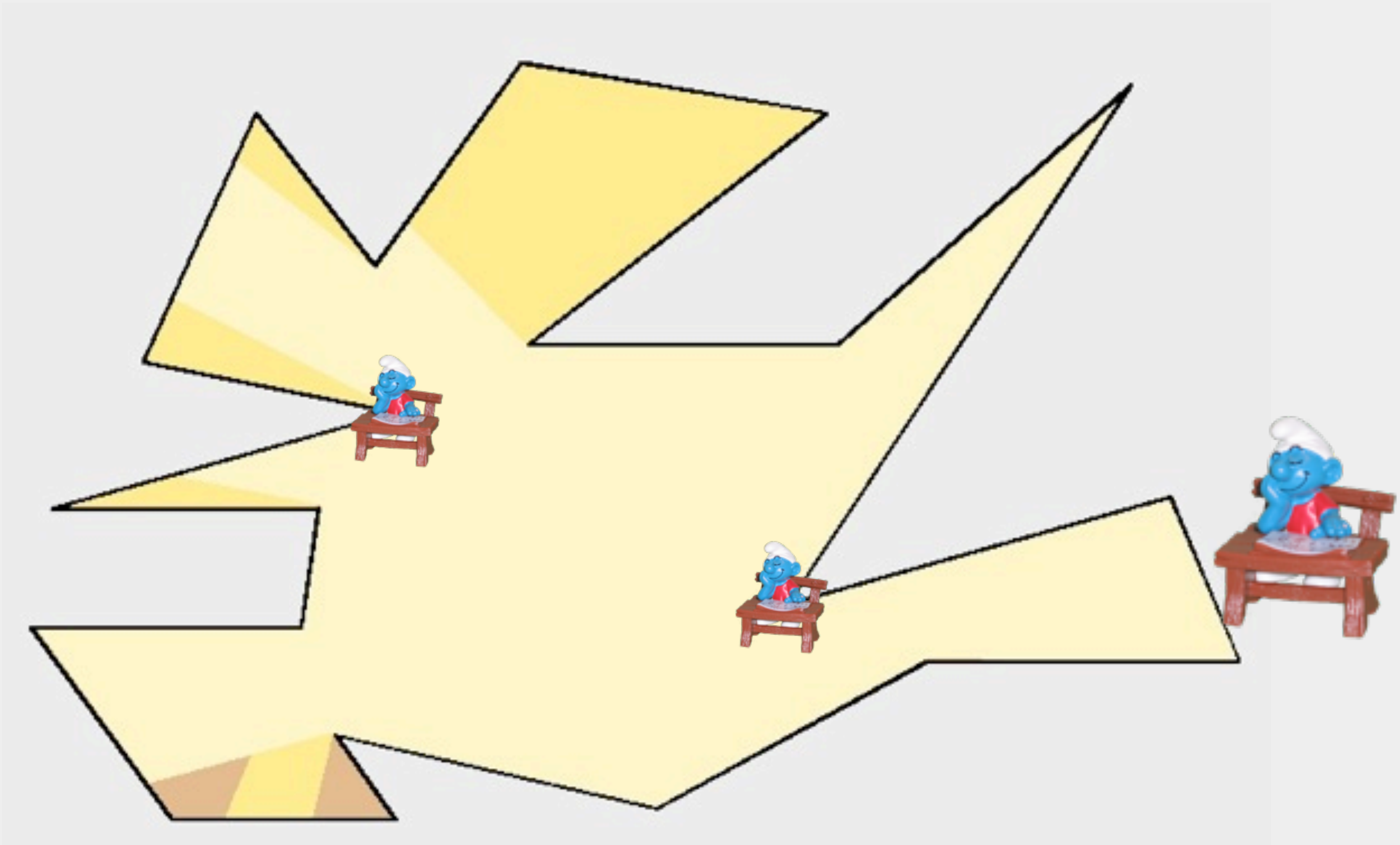
Online Searching with an Autonomous Robot.

Computational Geometry: Theory and Applications, 34 (2), 2006, pp. 102-115.

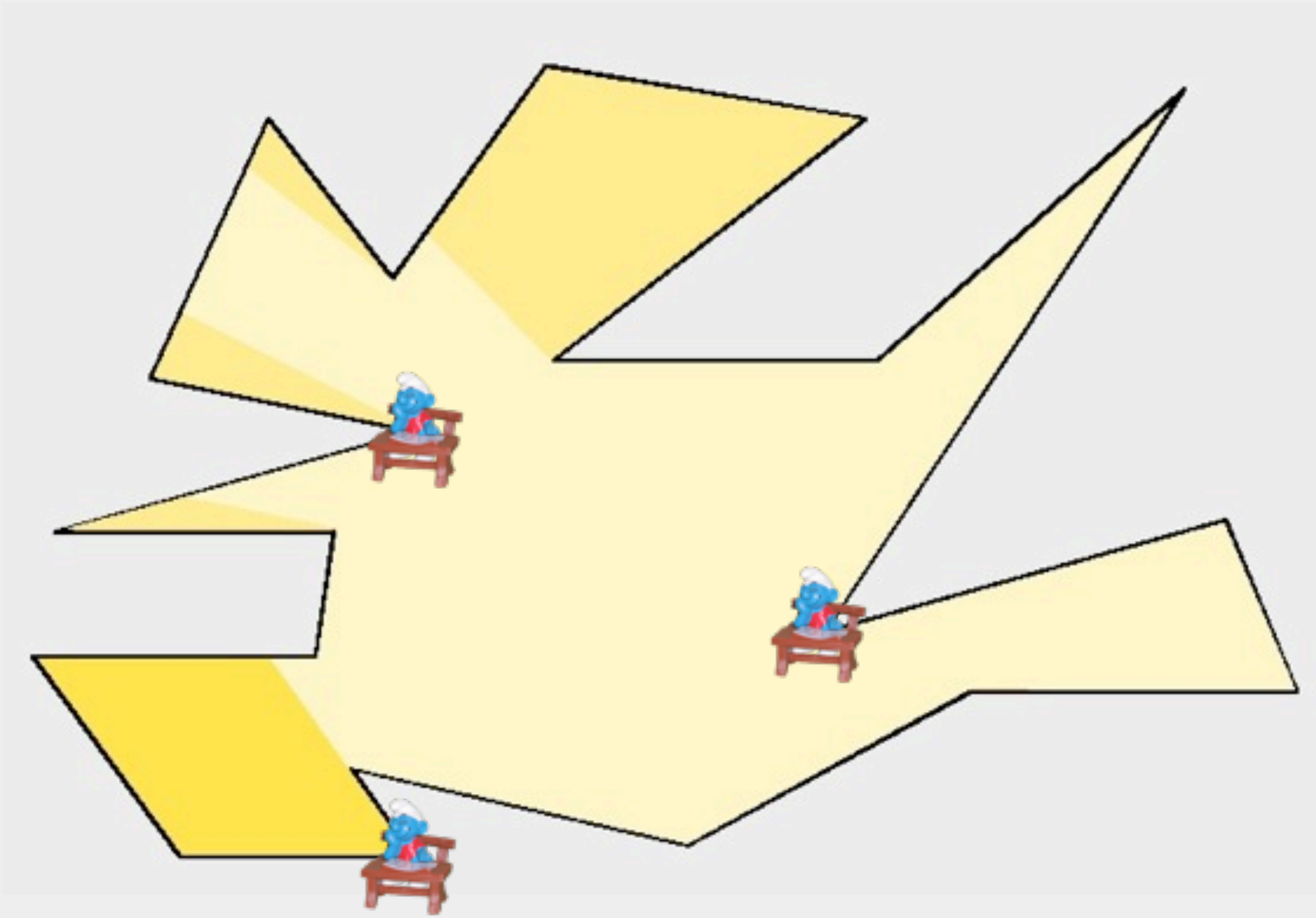
Art Gallery Problems



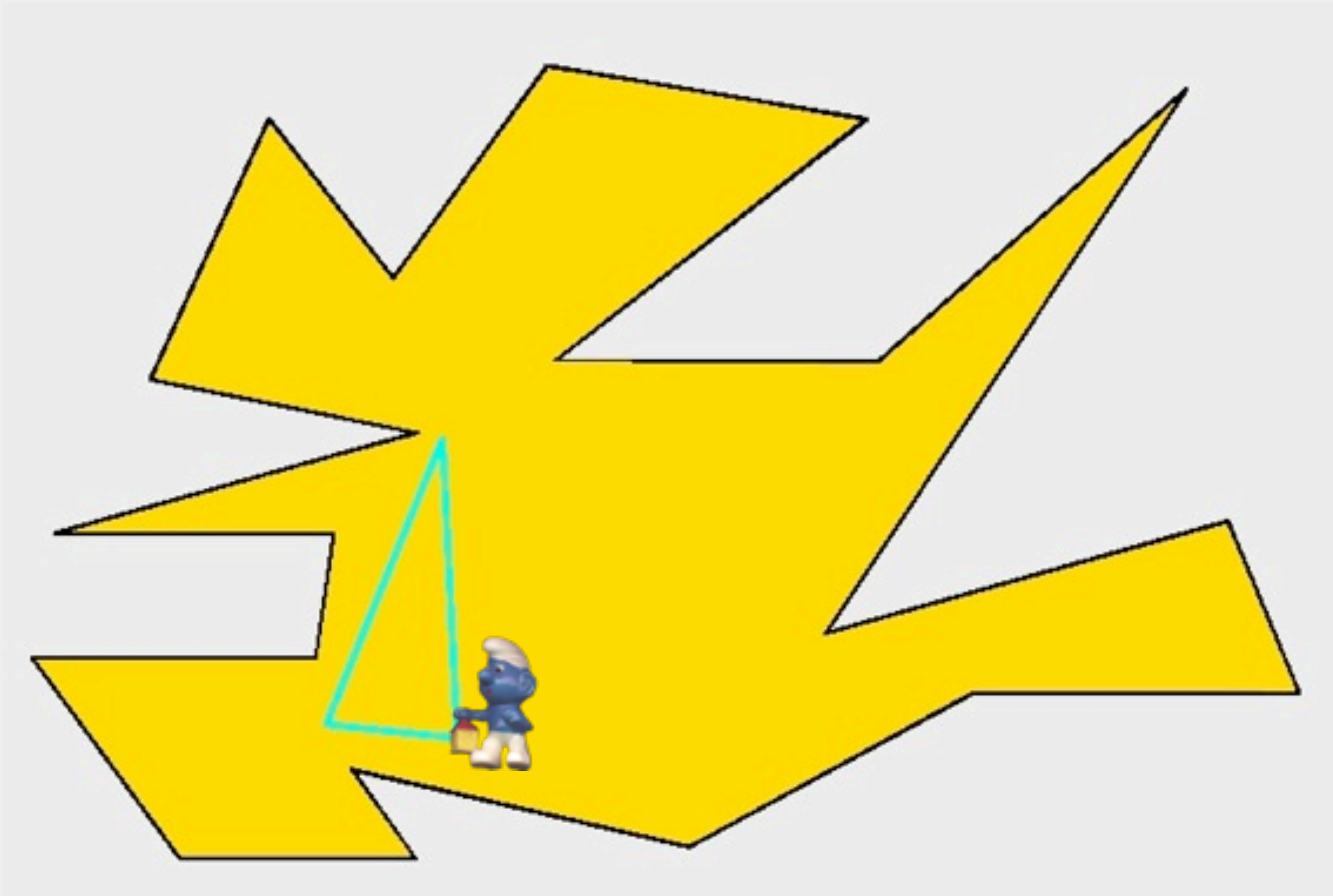
Art Gallery Problems



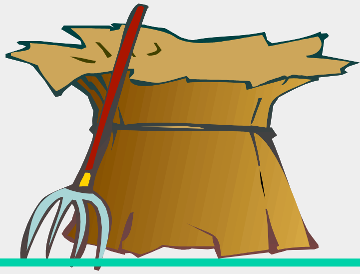
Art Gallery Problems

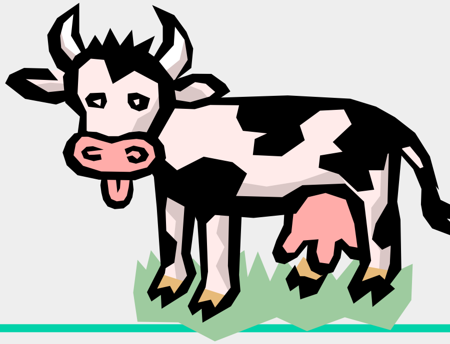


Watchman Problems

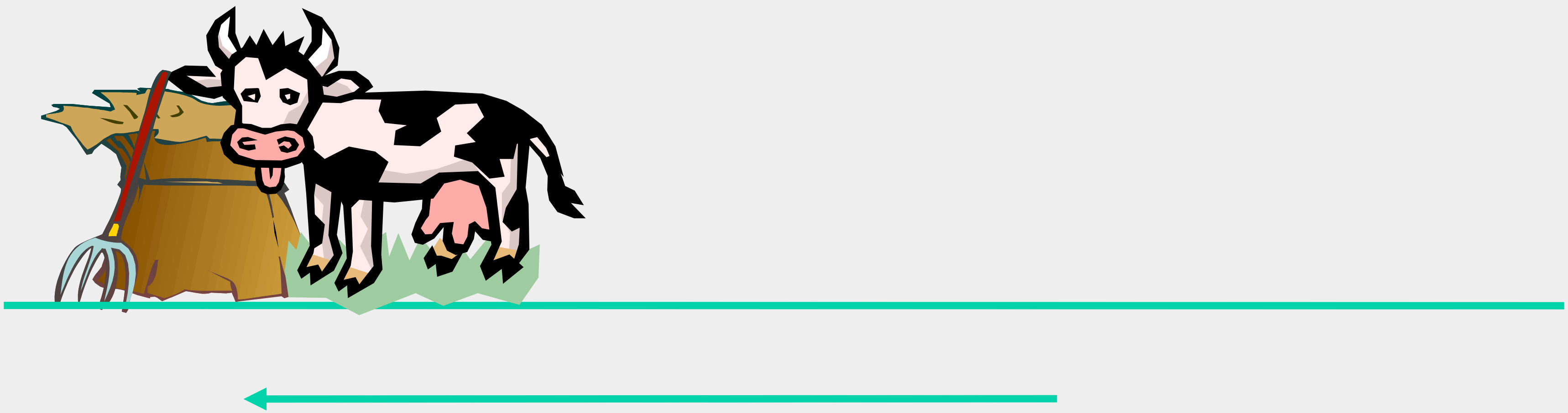


Online Searching

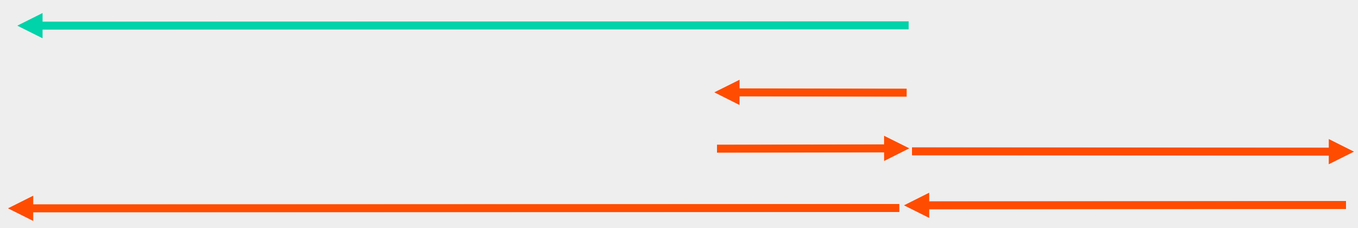




Online Searching



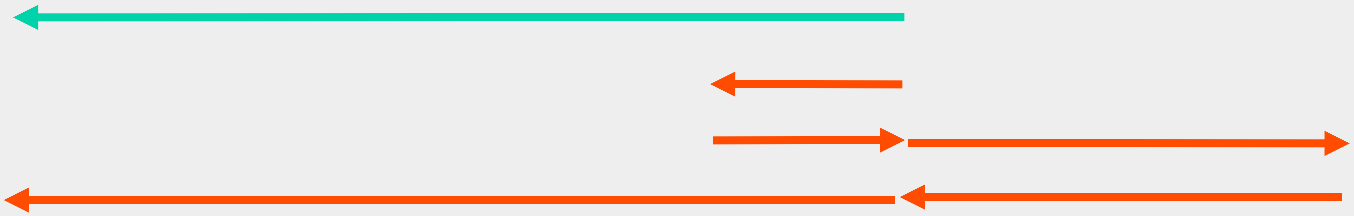
Online Searching



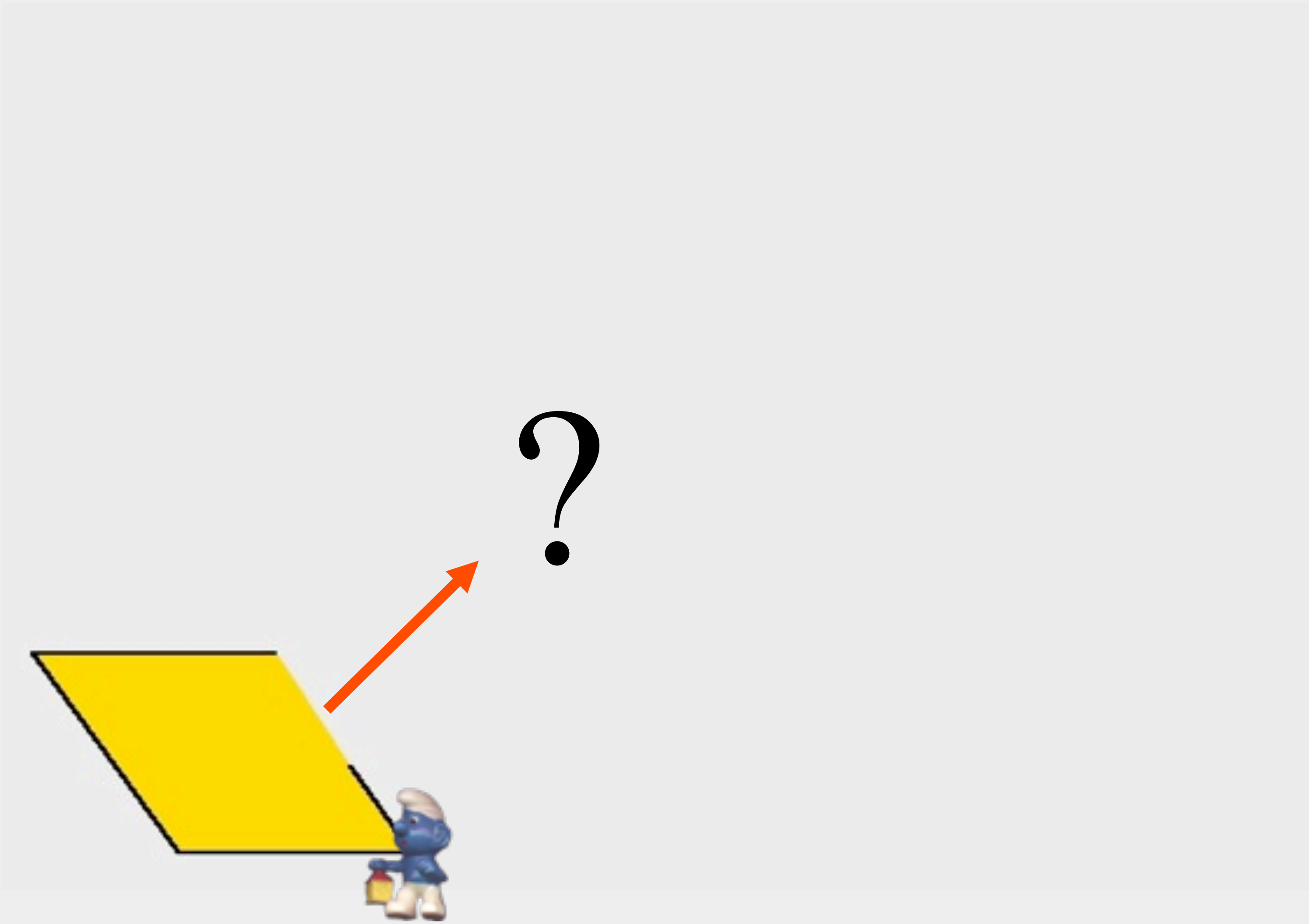
Online Searching

competitive ratio

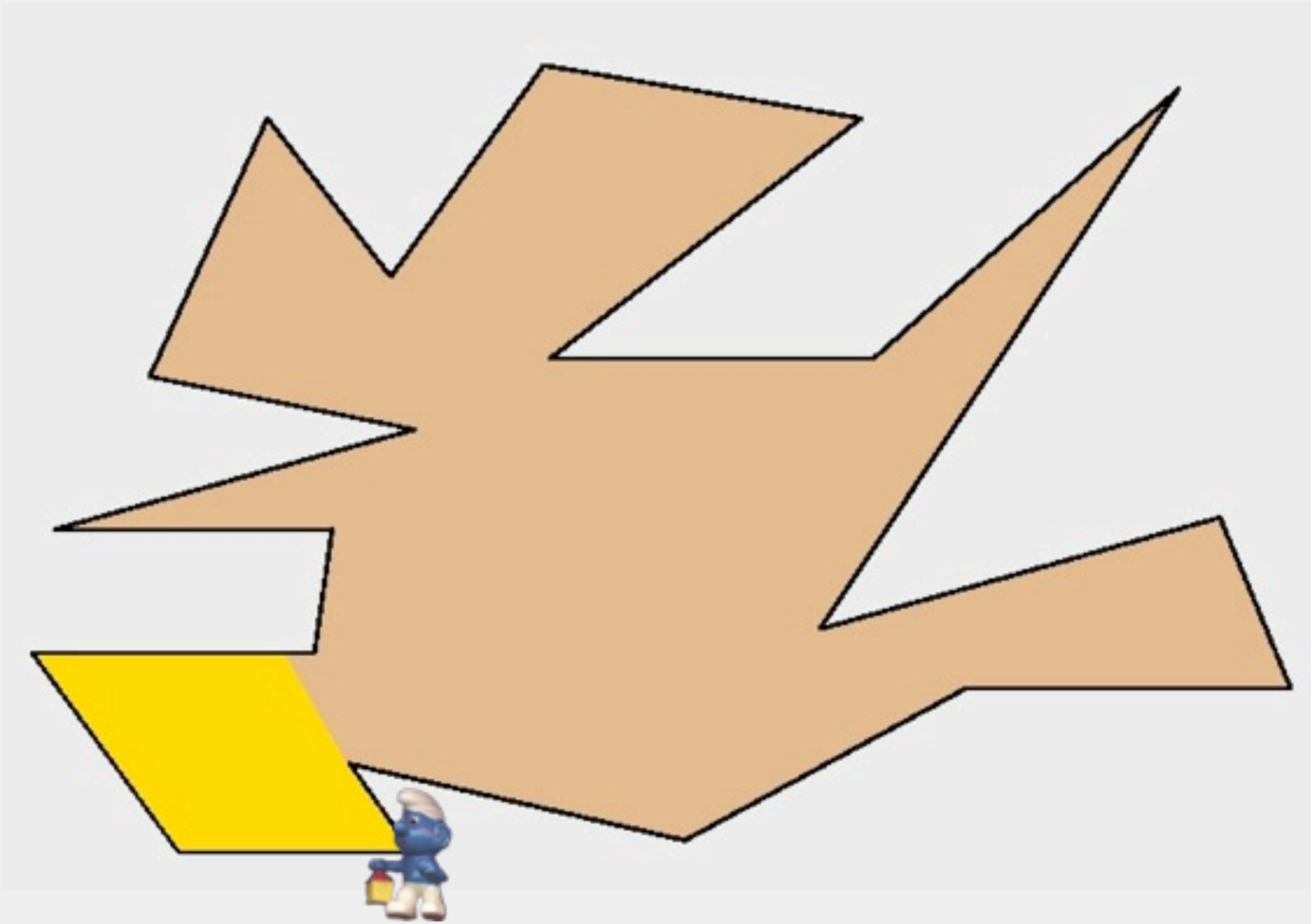
$$c = \sup \frac{\text{achieved time}}{\text{optimal time}}$$



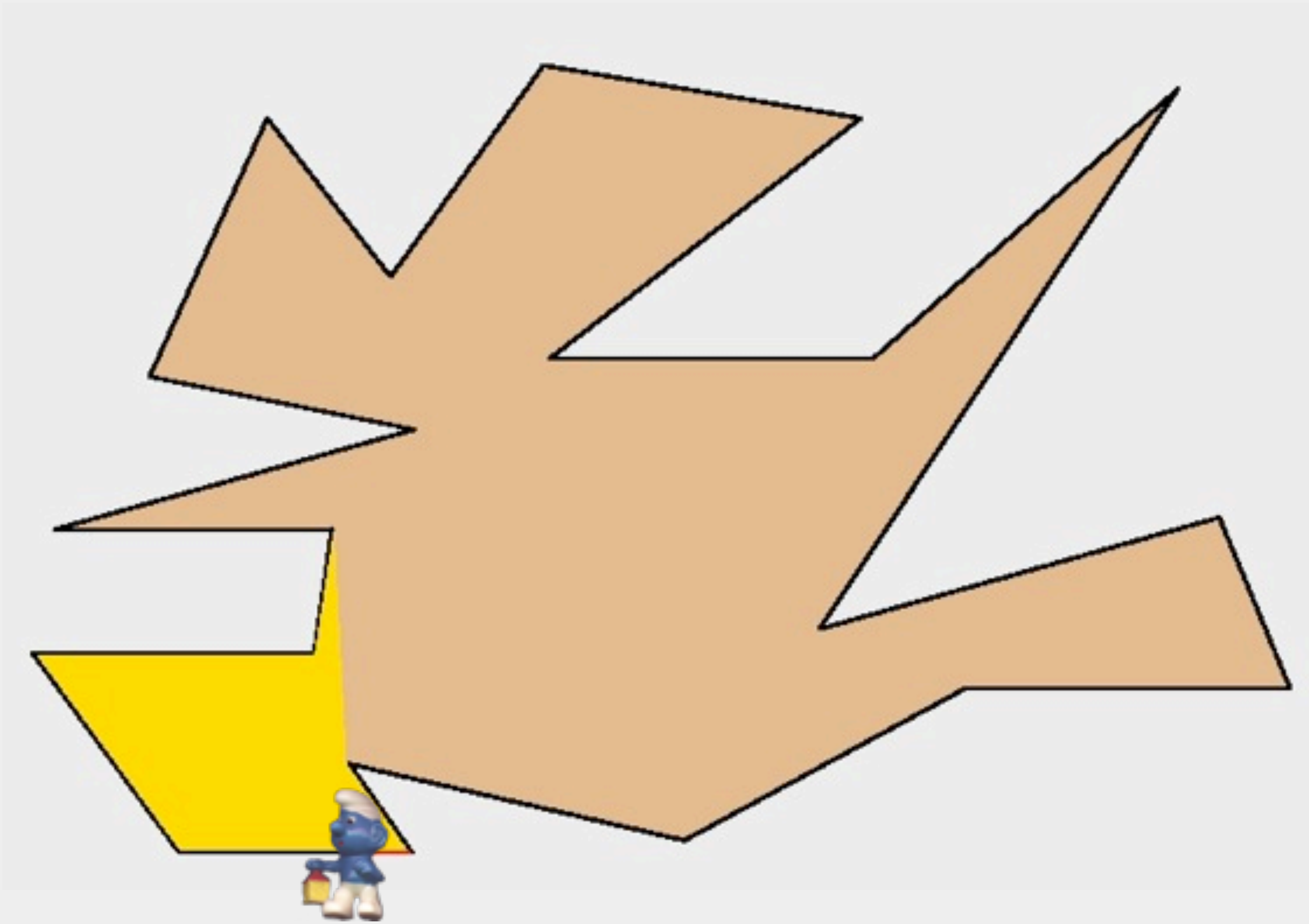
Online Exploration of Simple Polygons



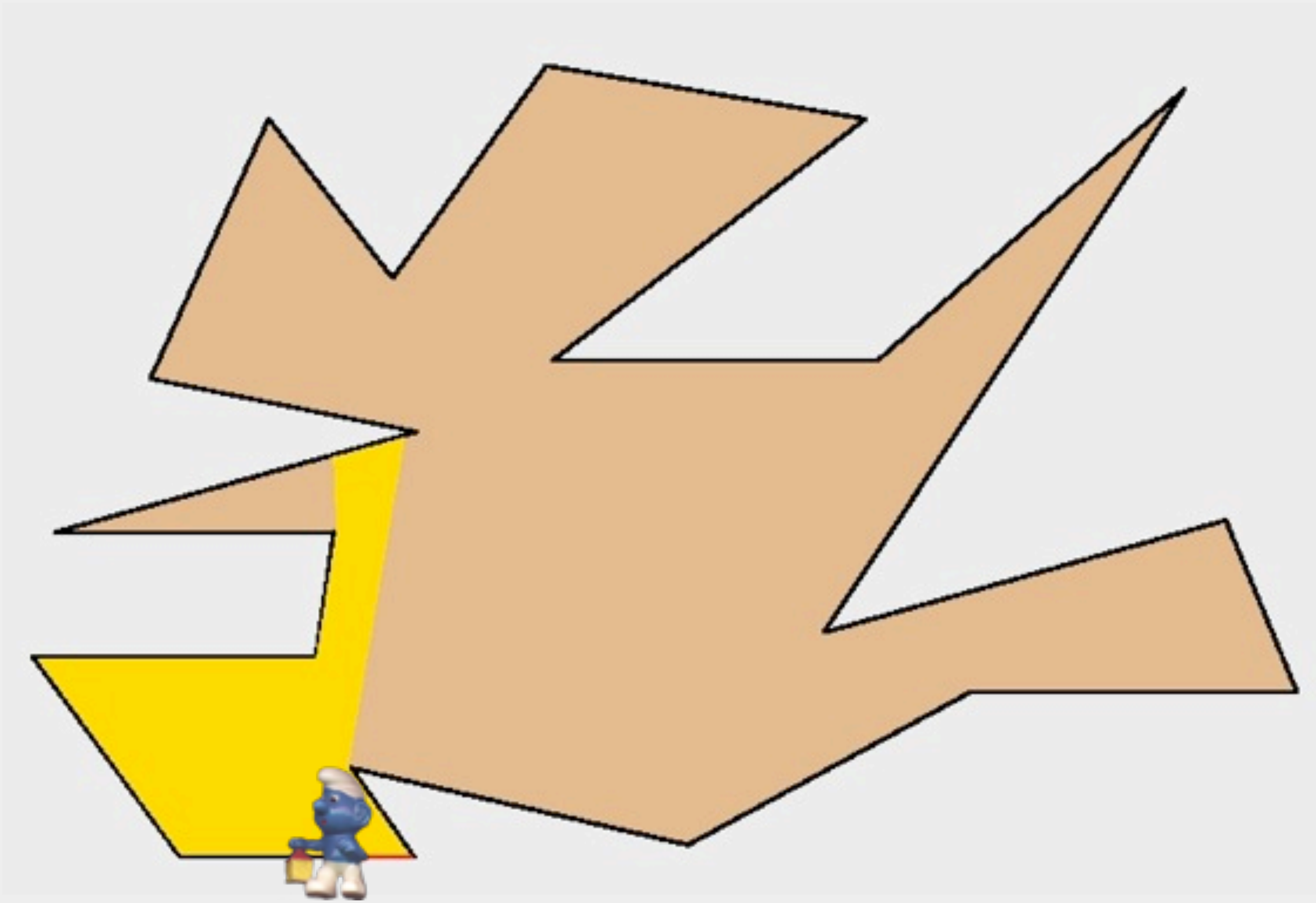
Online Exploration of Simple Polygons



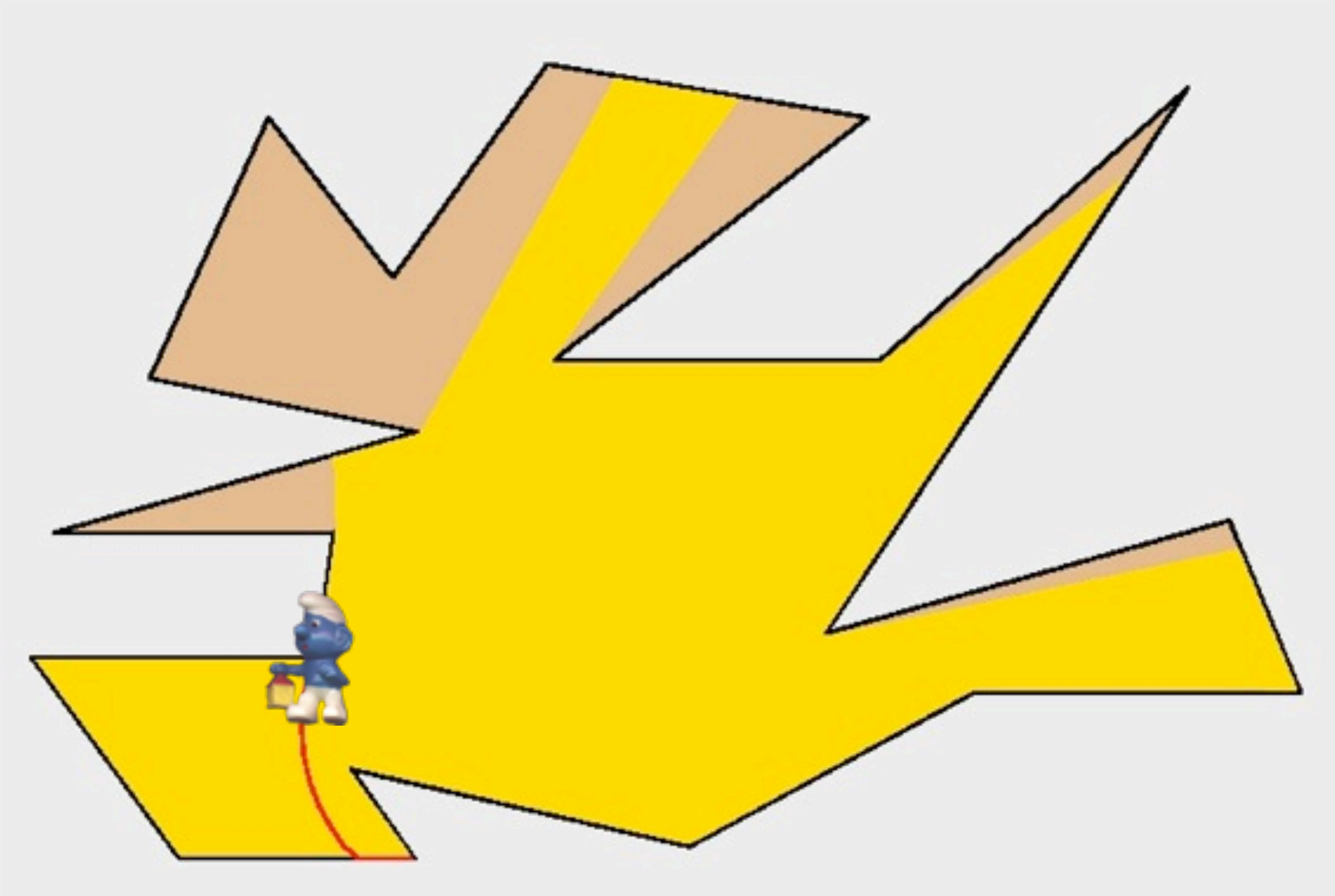
Online Exploration of Simple Polygons



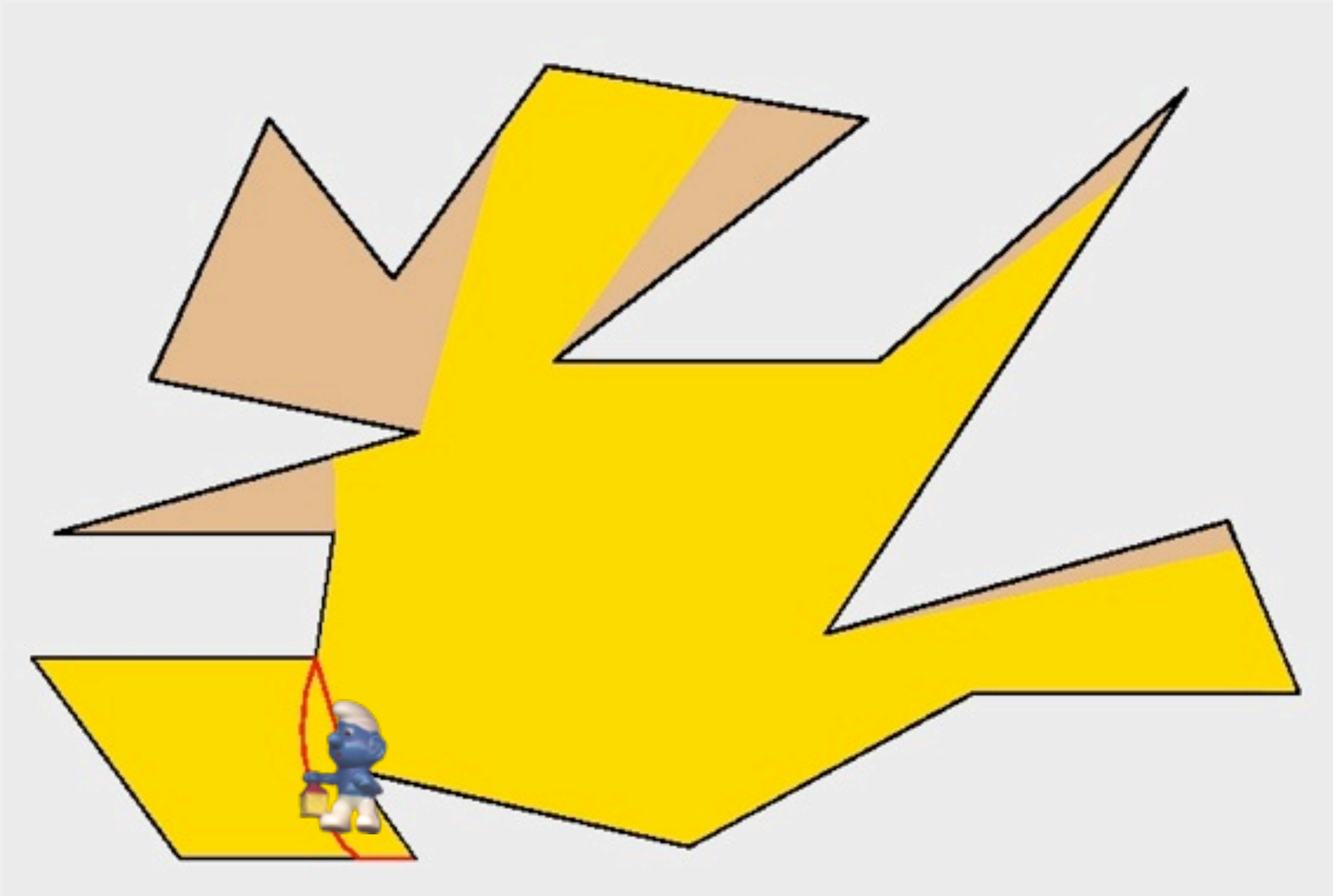
Online Exploration of Simple Polygons



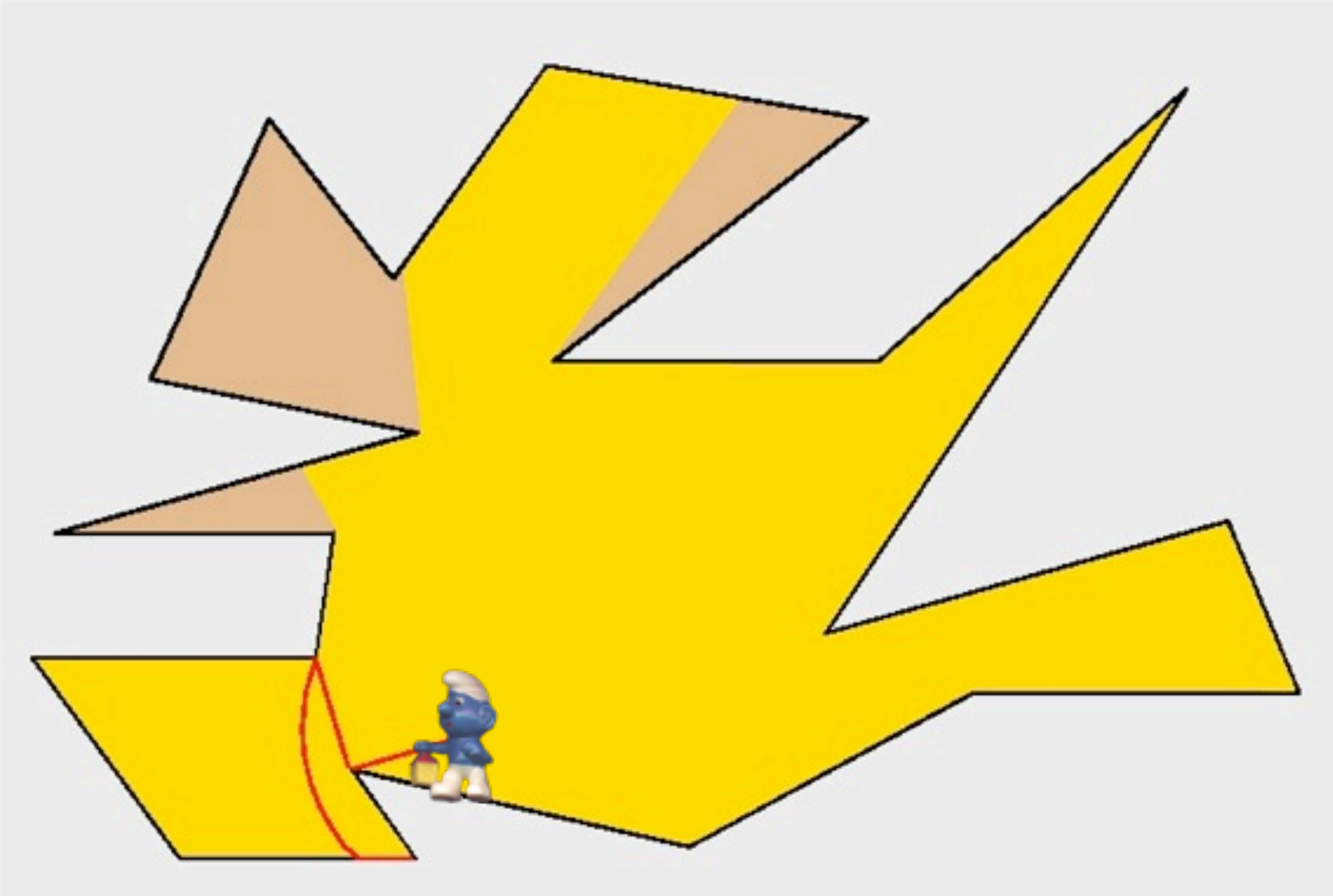
Online Exploration of Simple Polygons



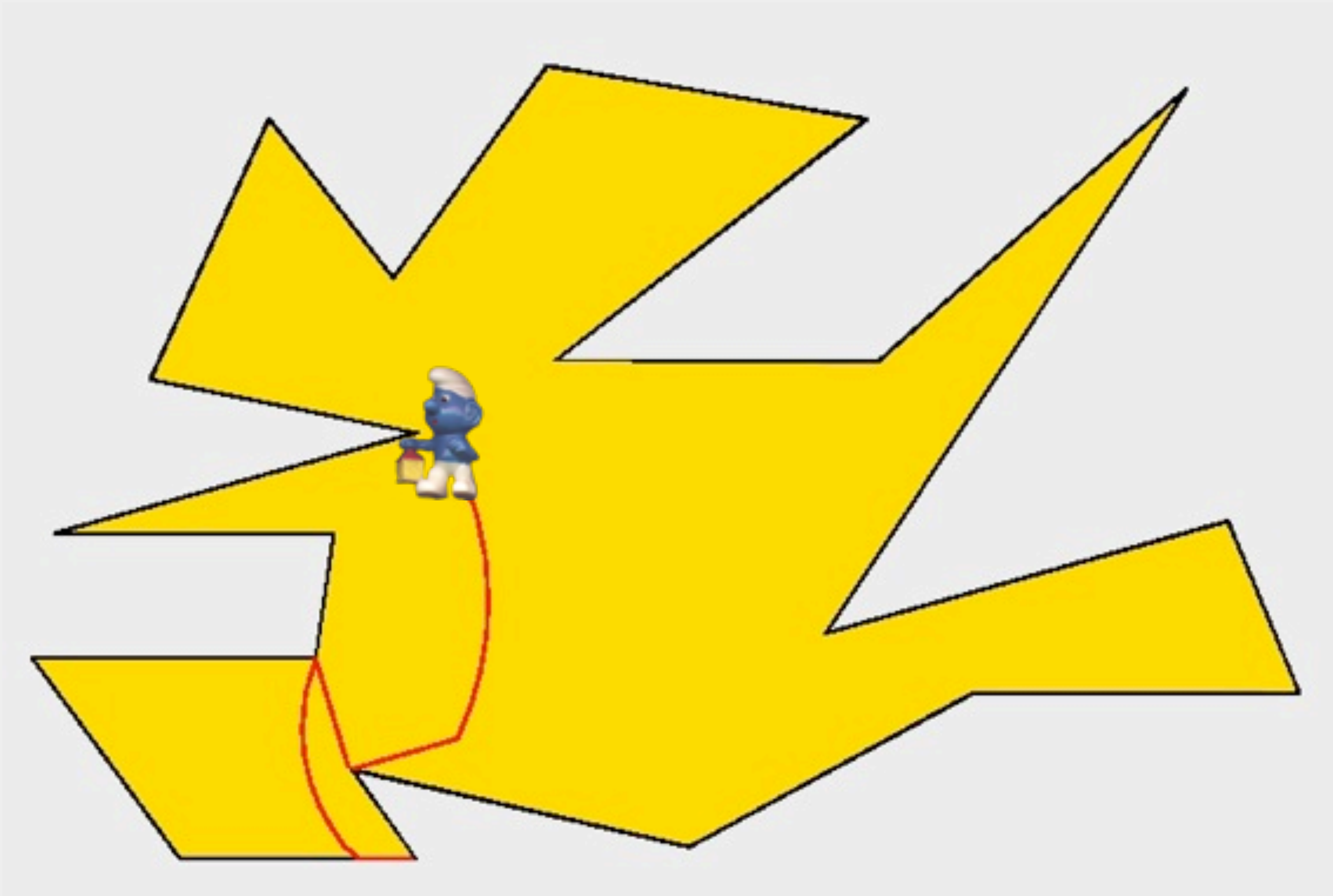
Online Exploration of Simple Polygons



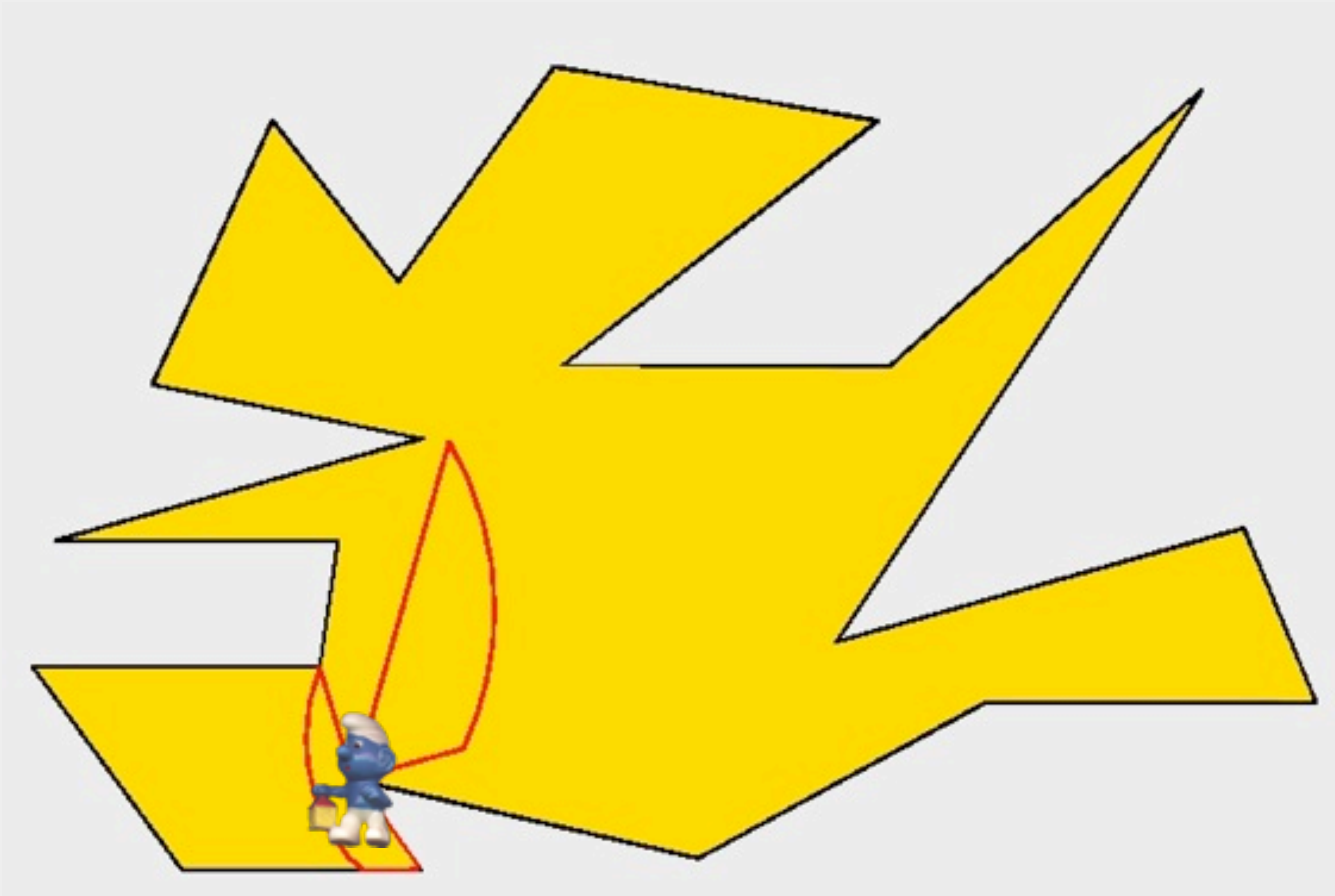
Online Exploration of Simple Polygons



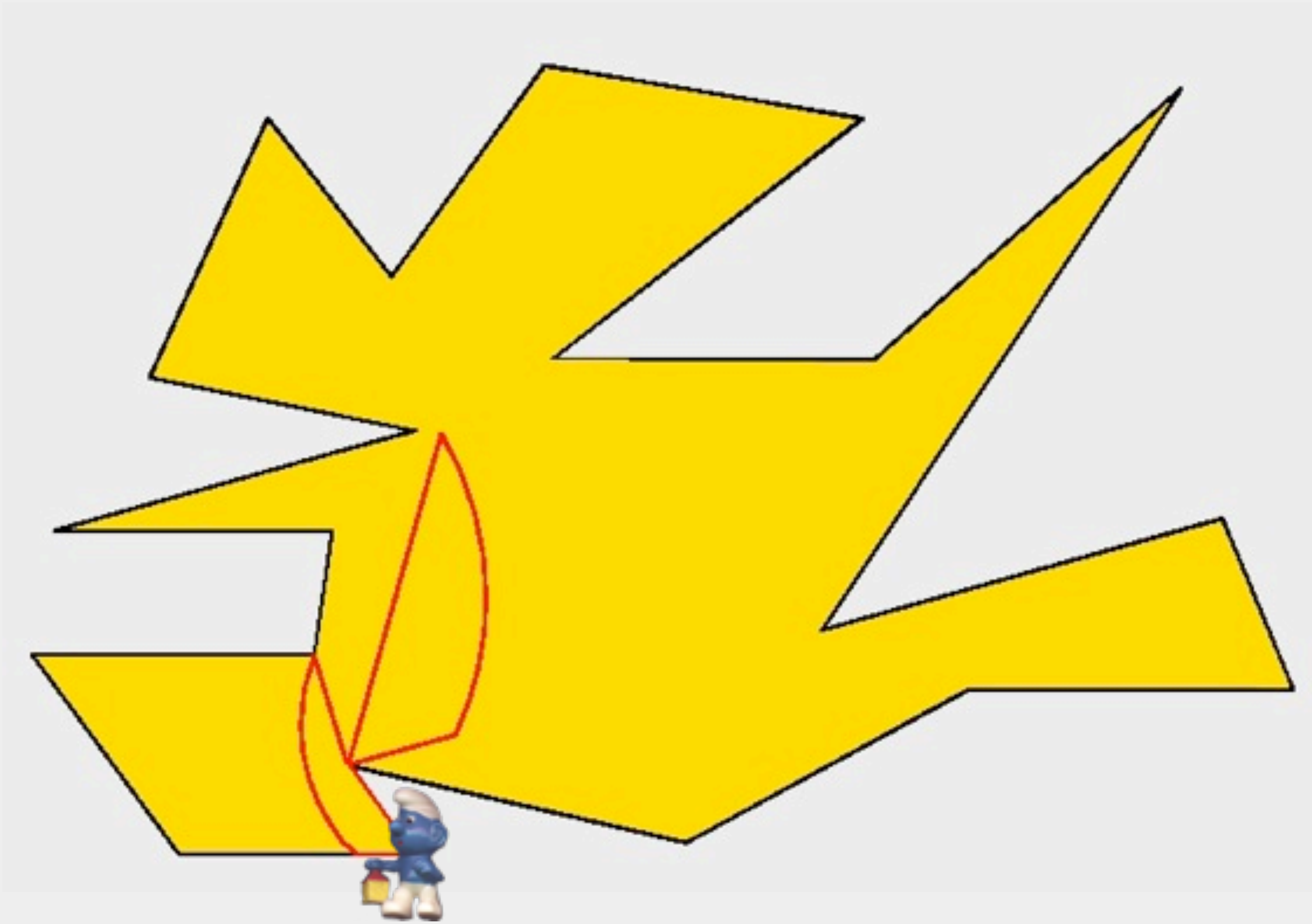
Online Exploration of Simple Polygons



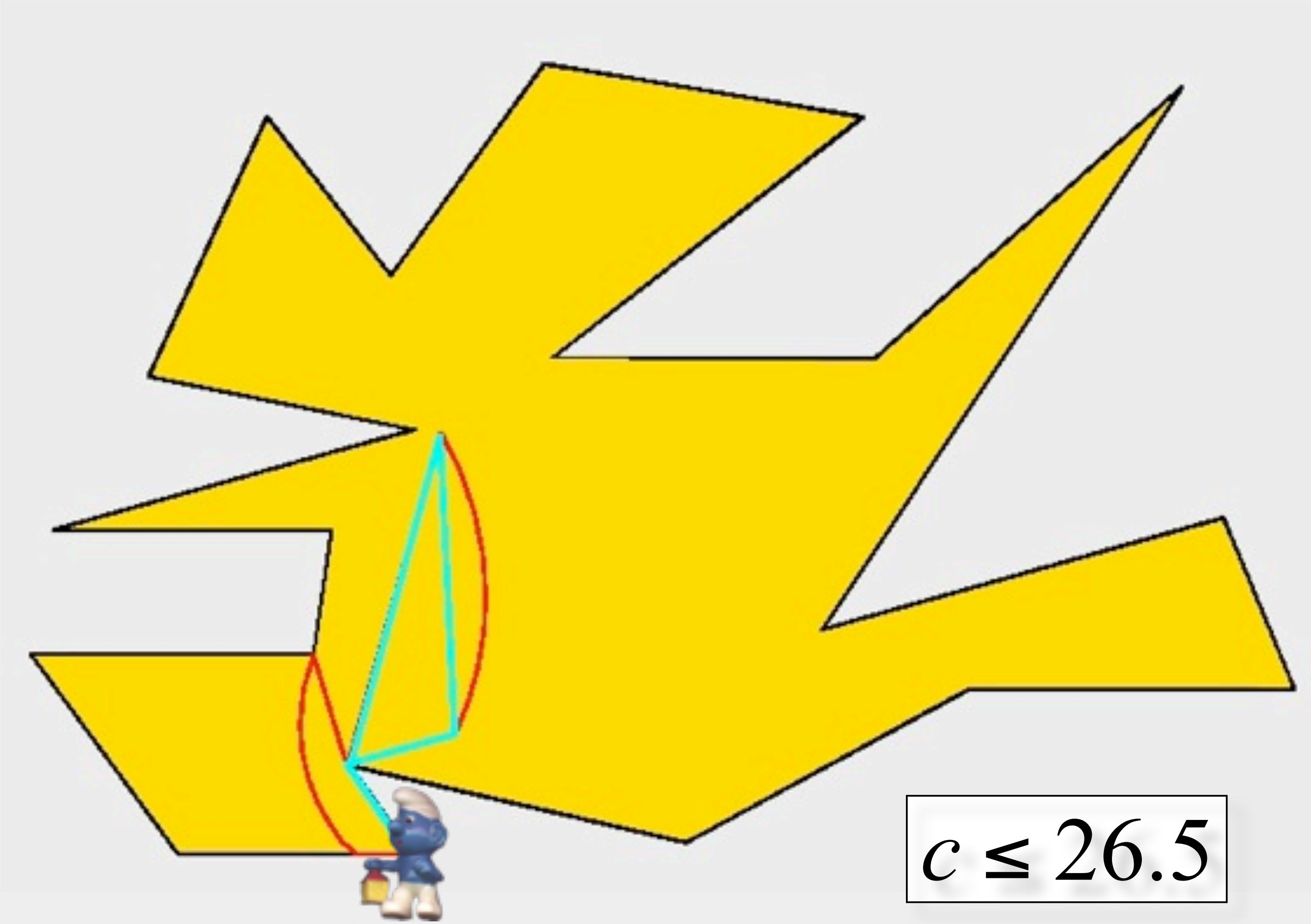
Online Exploration of Simple Polygons



Online Exploration of Simple Polygons

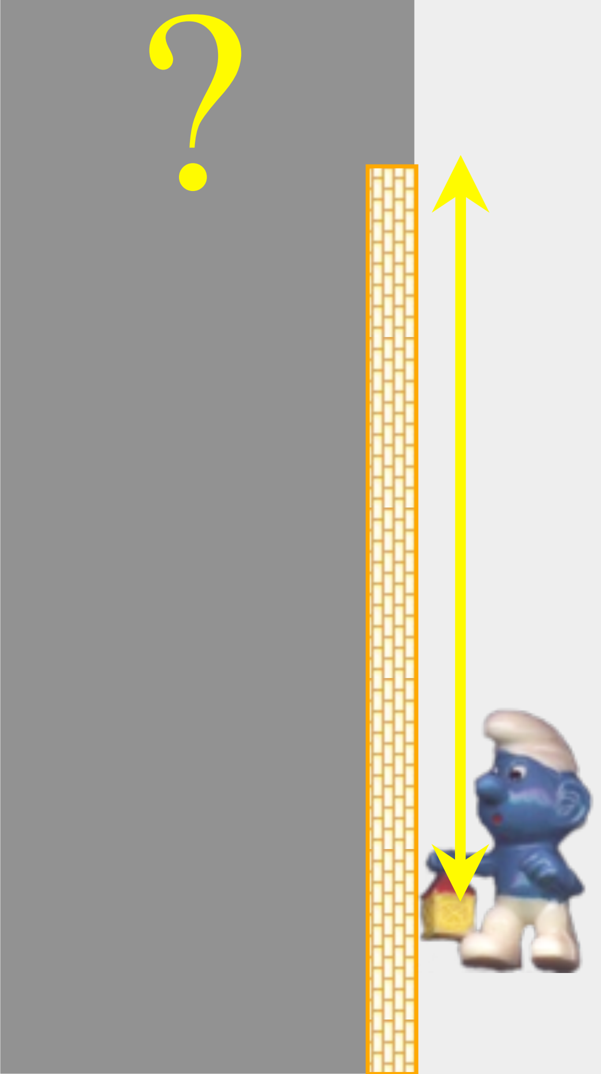


Online Exploration of Simple Polygons

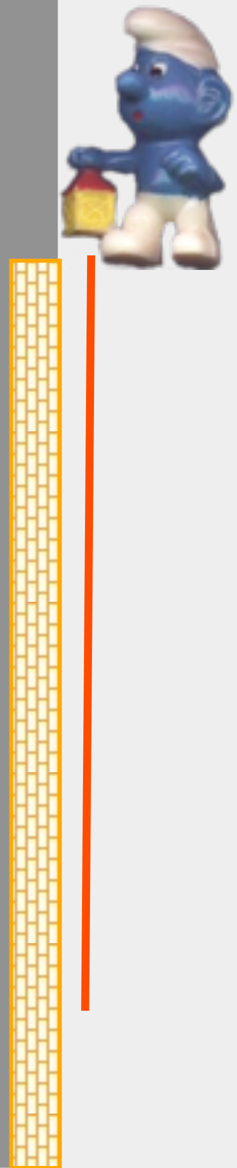


$c \leq 26.5$

How to Look Around a Corner

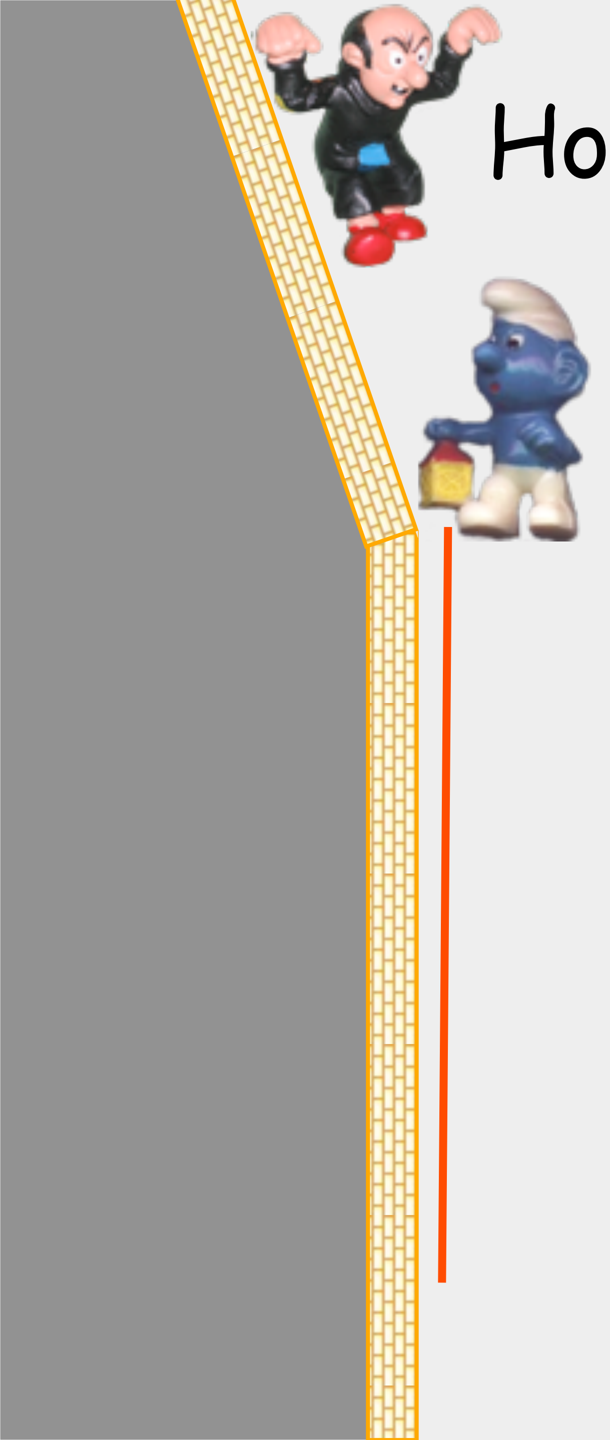


How to Look Around a Corner



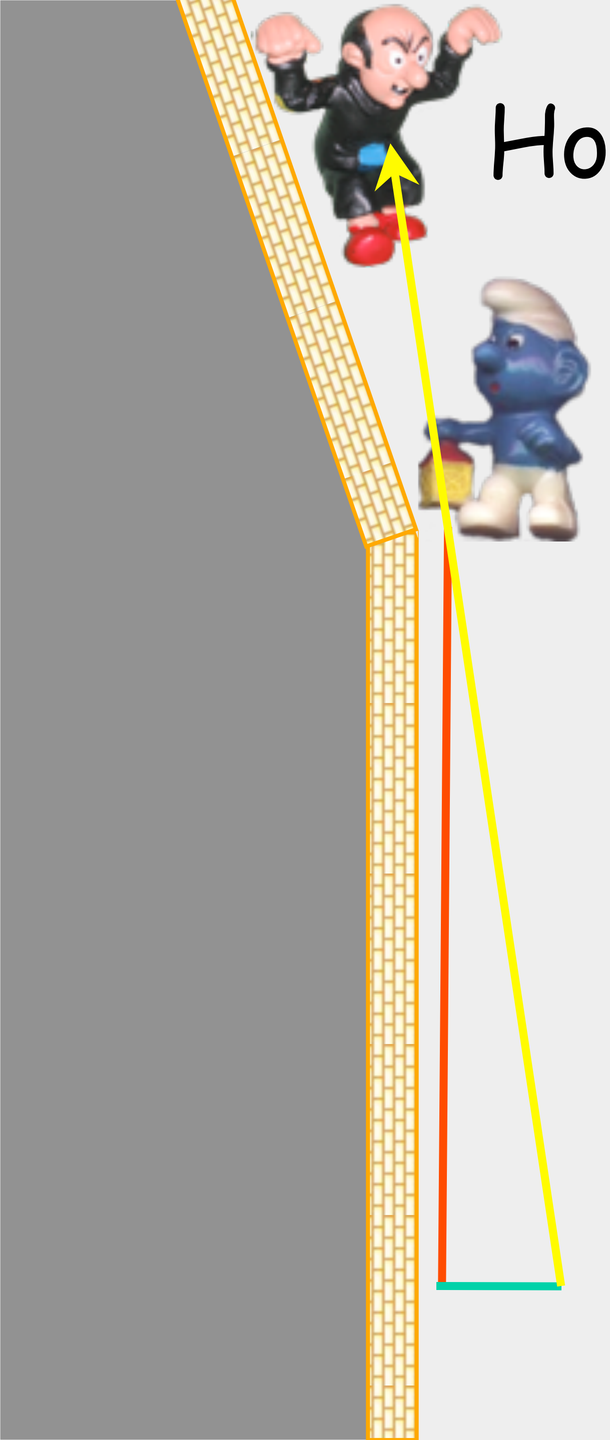
achieved = d

How to Look Around a Corner



achieved = d

How to Look Around a Corner

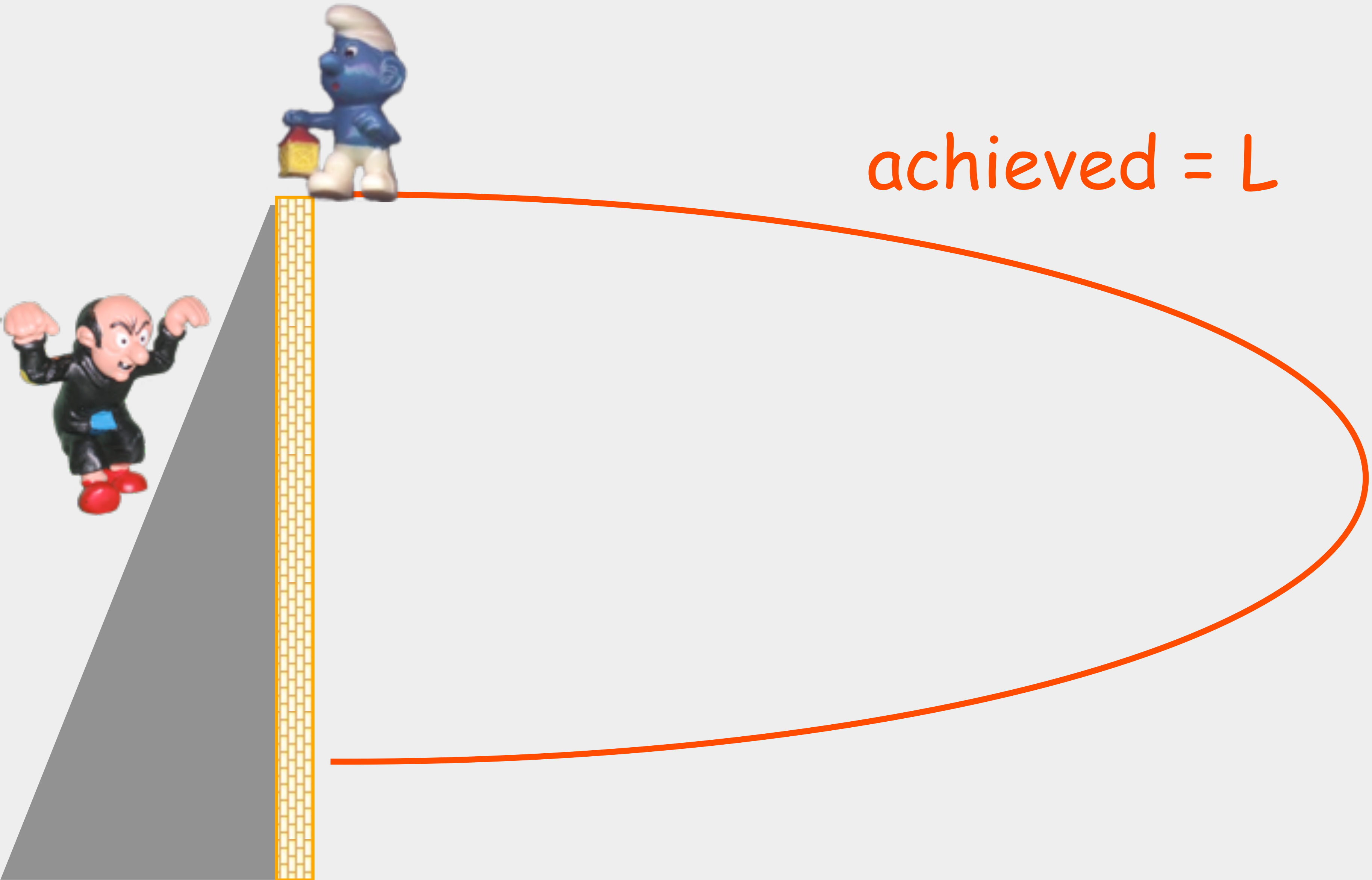


achieved = d

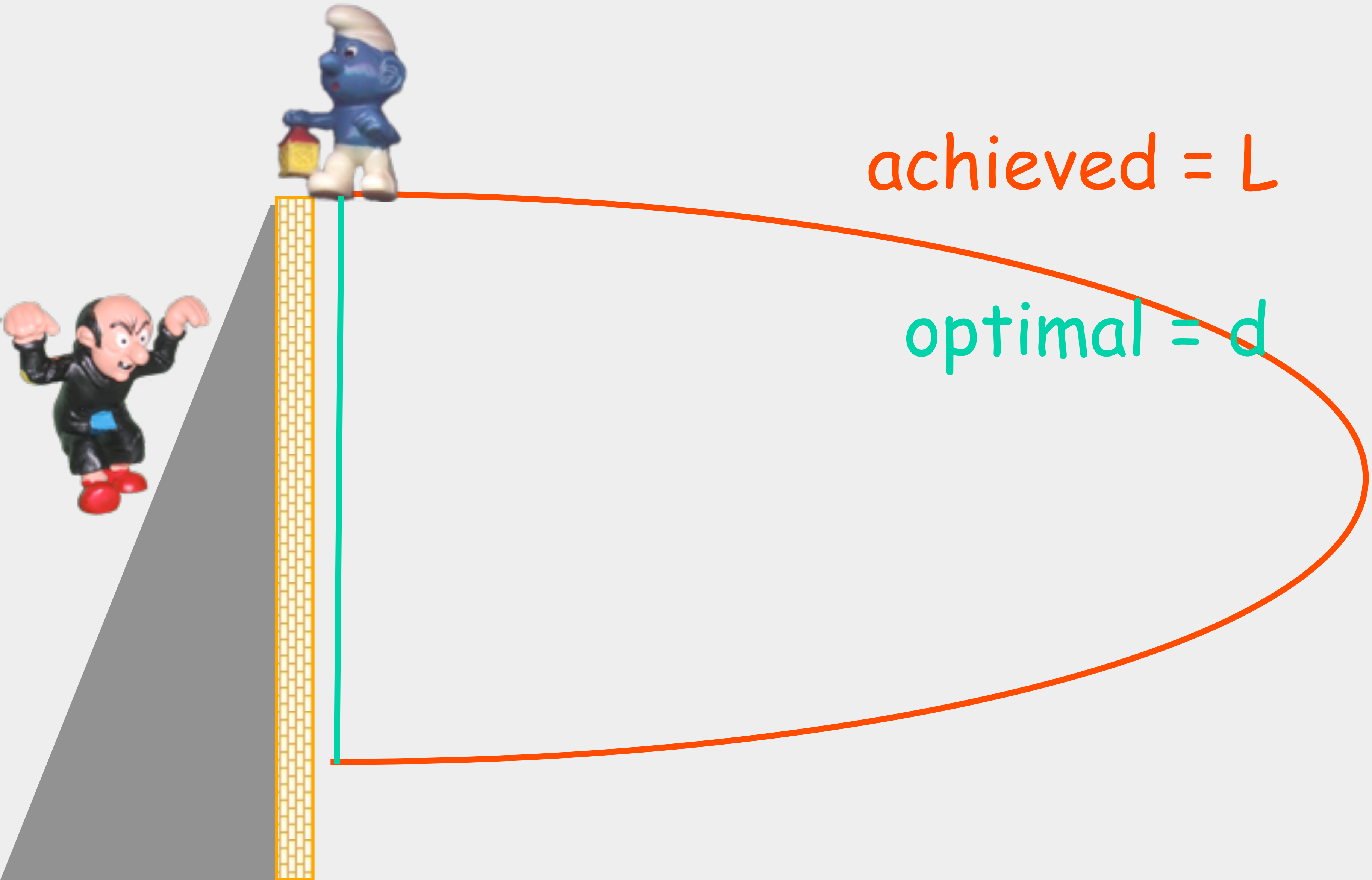
optimal = ε

$$c \to \infty$$

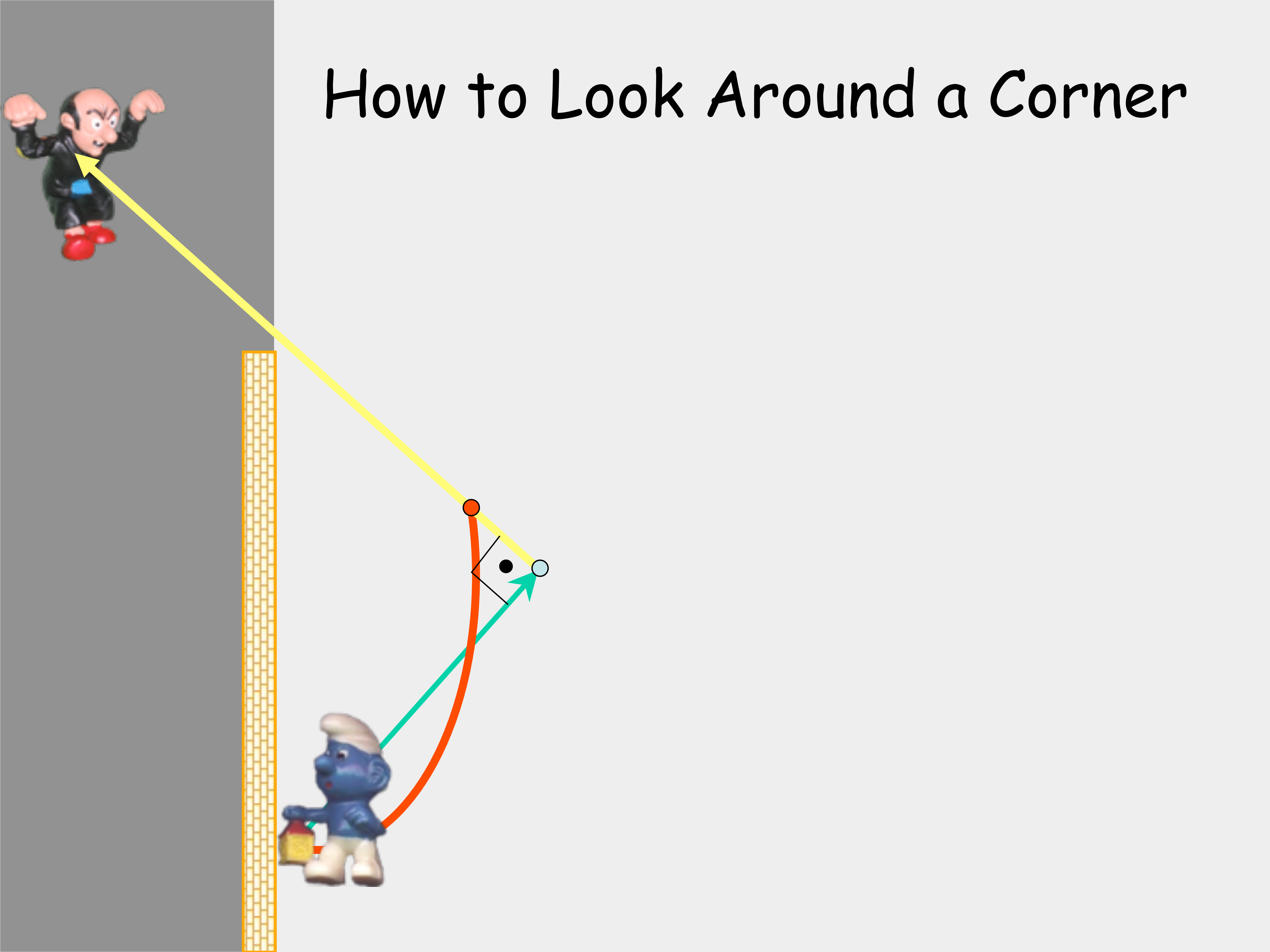
How to Look Around a Corner



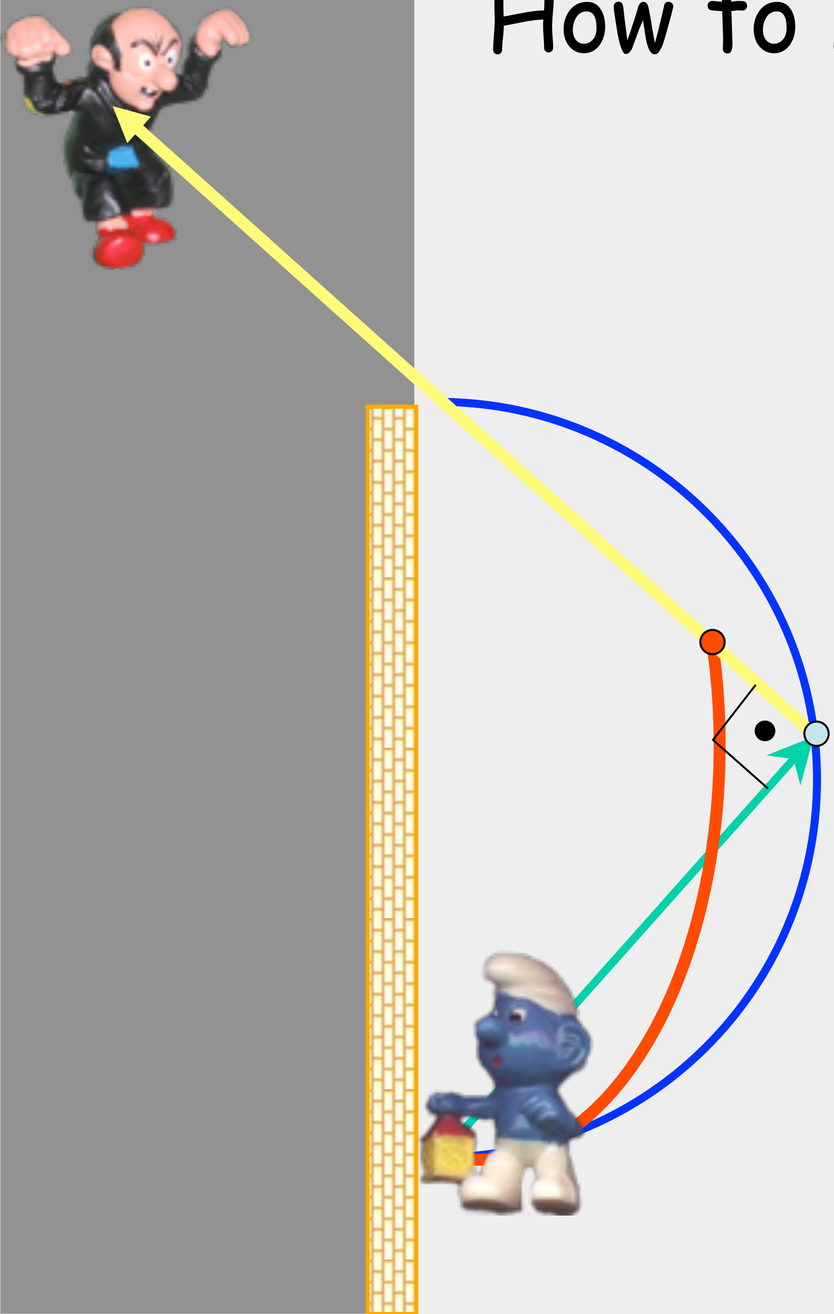
How to Look Around a Corner



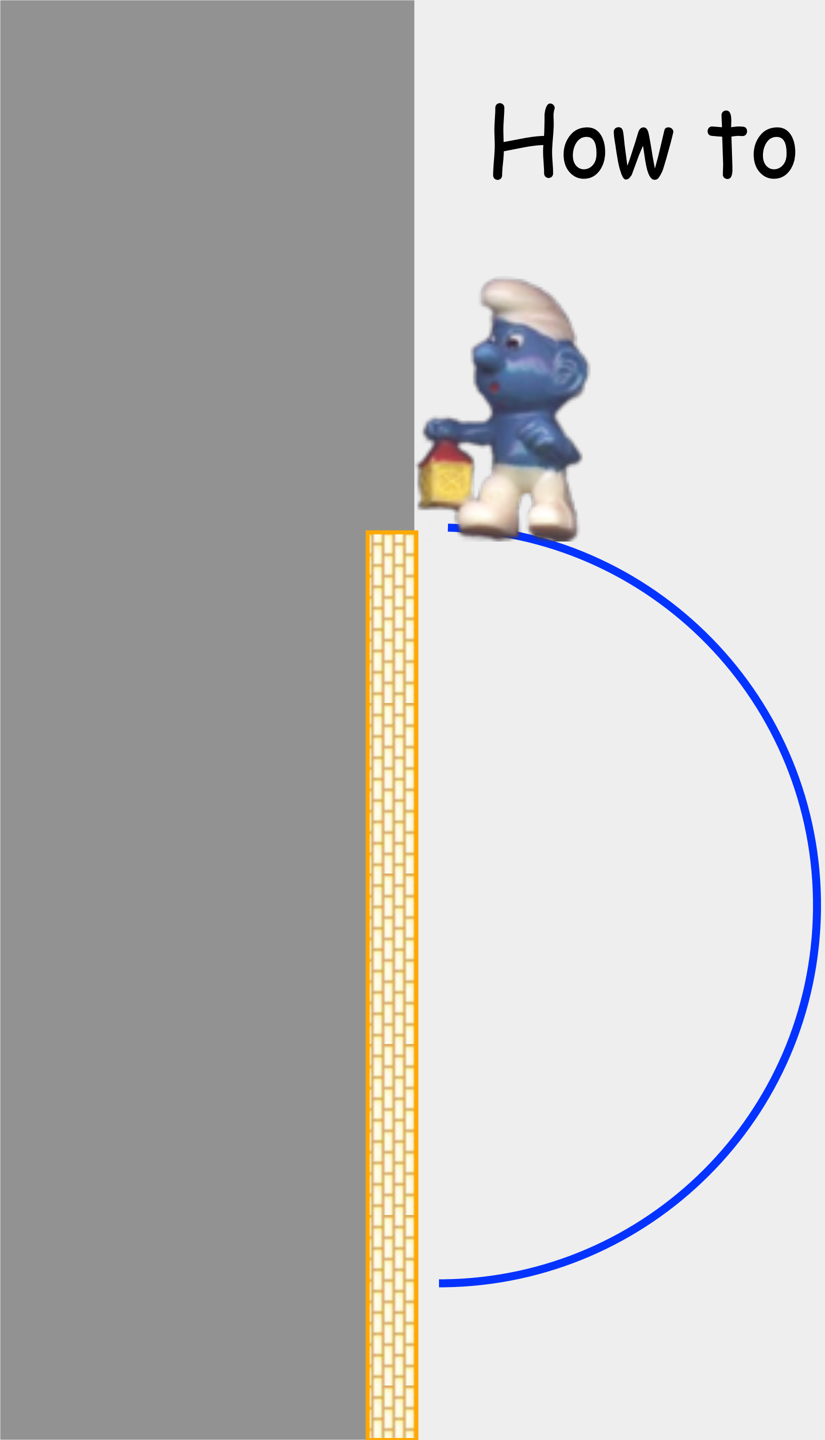
How to Look Around a Corner



How to Look Around a Corner



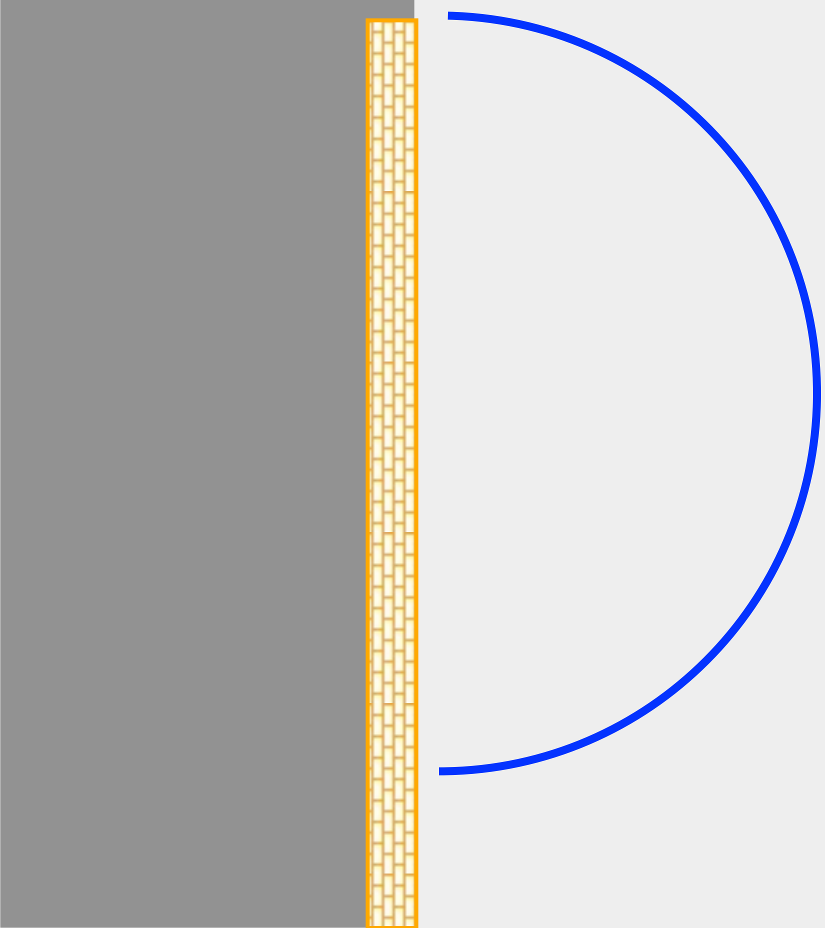
How to Look Around a Corner



achieved = $\pi d/2$

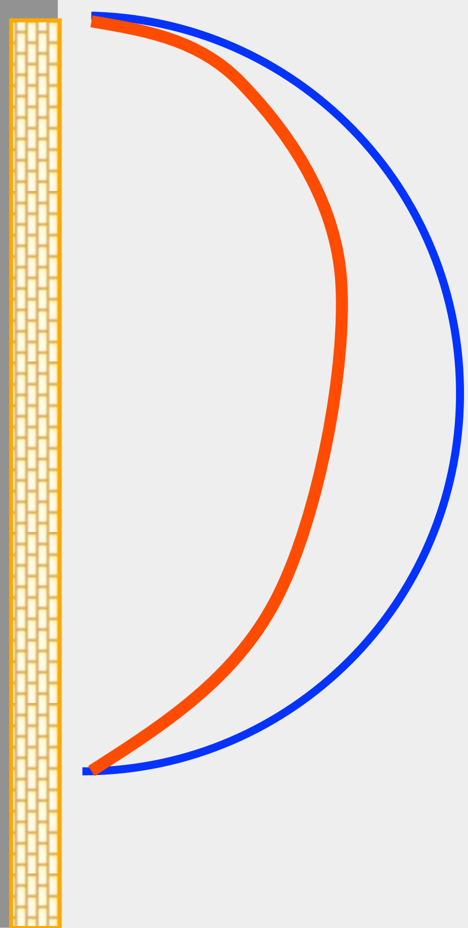
$$c = \pi / 2 = 1.57\ldots$$

How to Look Around a Corner



$$r'(\varphi) = -\sqrt{c^2 \cos\varphi - r^2(\varphi)}$$

How to Look Around a Corner



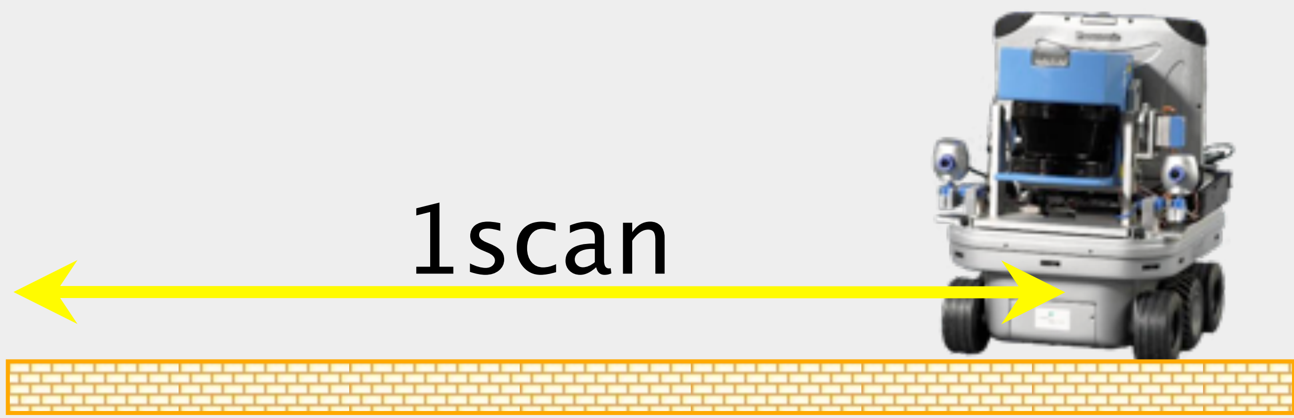
$$r'(\varphi) = -\sqrt{c^2 \cos\varphi - r^2(\varphi)}$$

$$c = 1.21\ldots$$

An Autonomous Robot

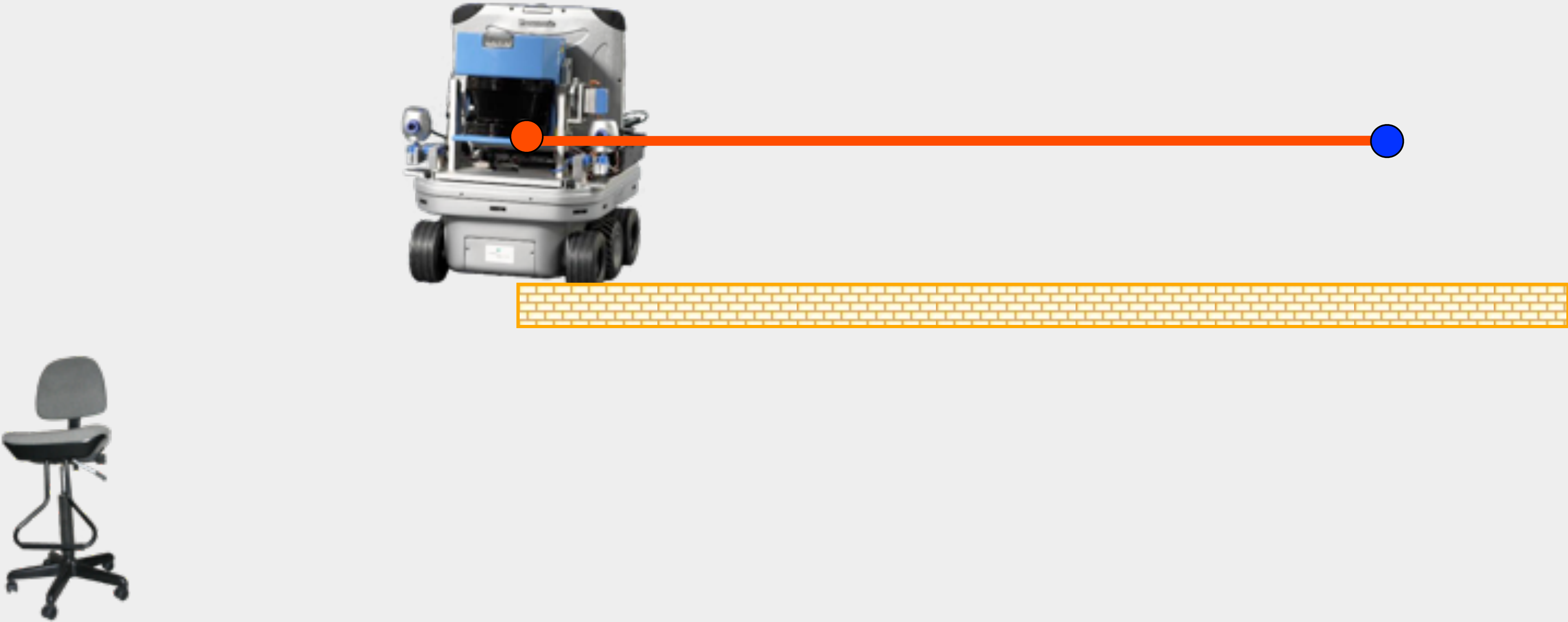
An Autonomous Robot

Short Distances



Short Distances

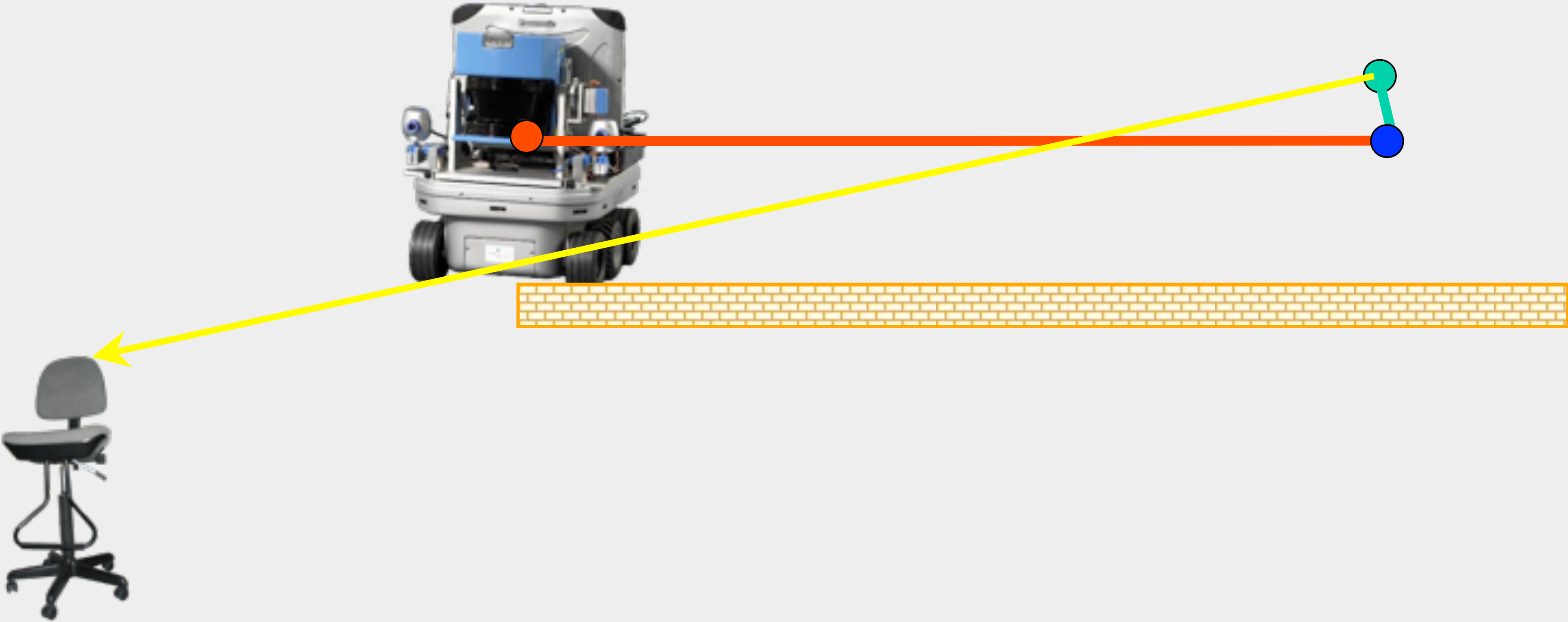
achieved: 2 sc



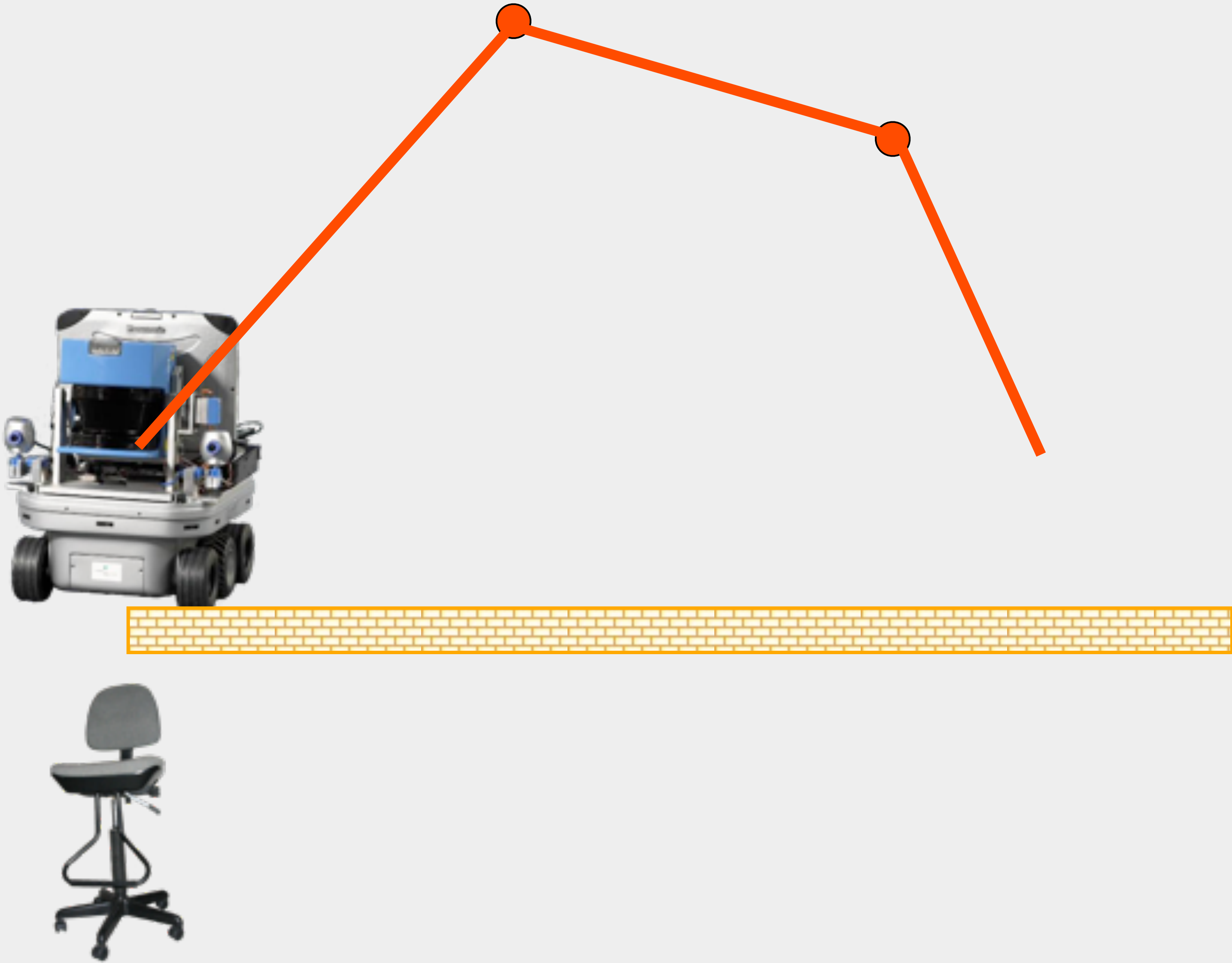
Short Distances

achieved: 2 sc

optimal: (1+ε) sc

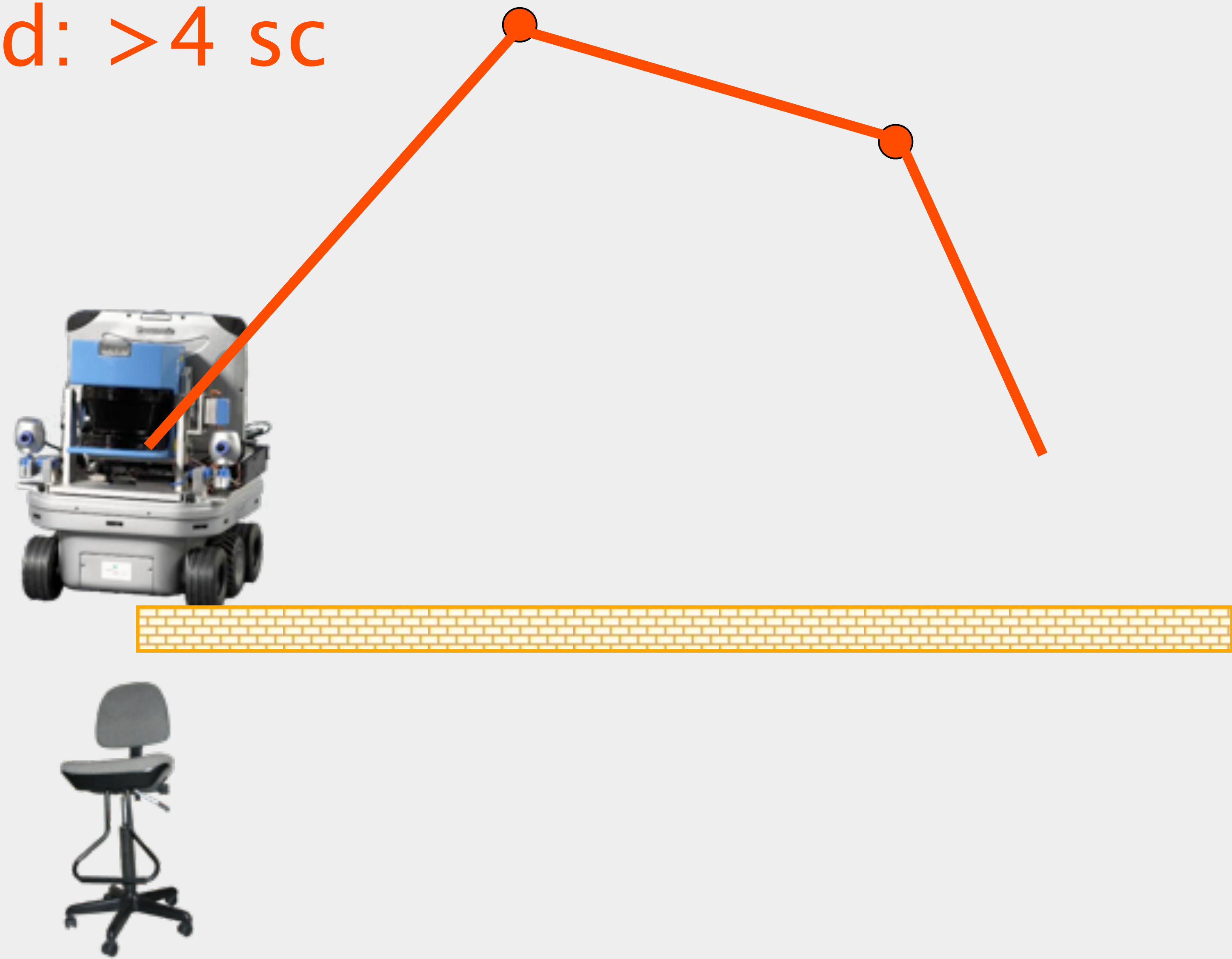


Short Distances



Short Distances

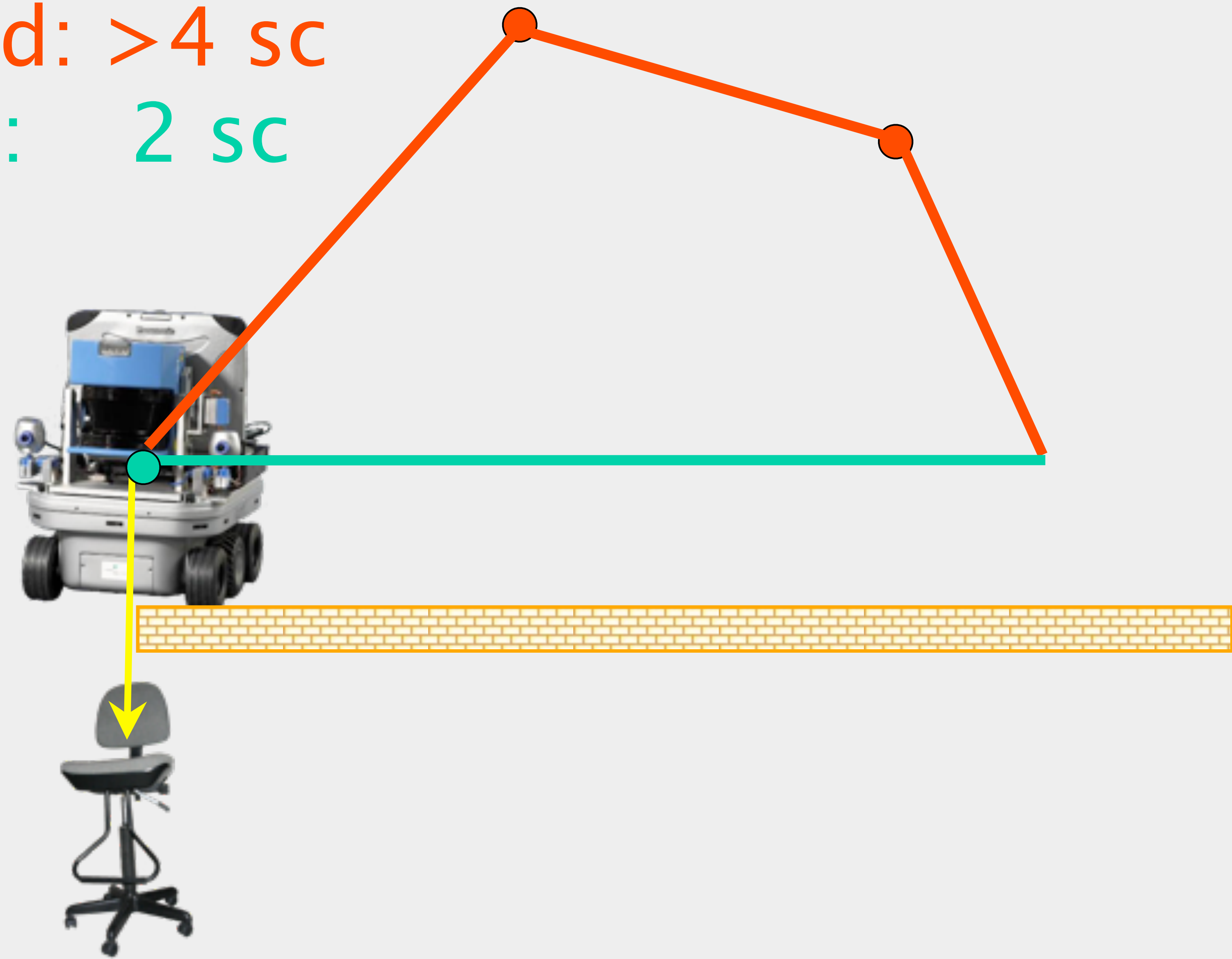
achieved: >4 sc



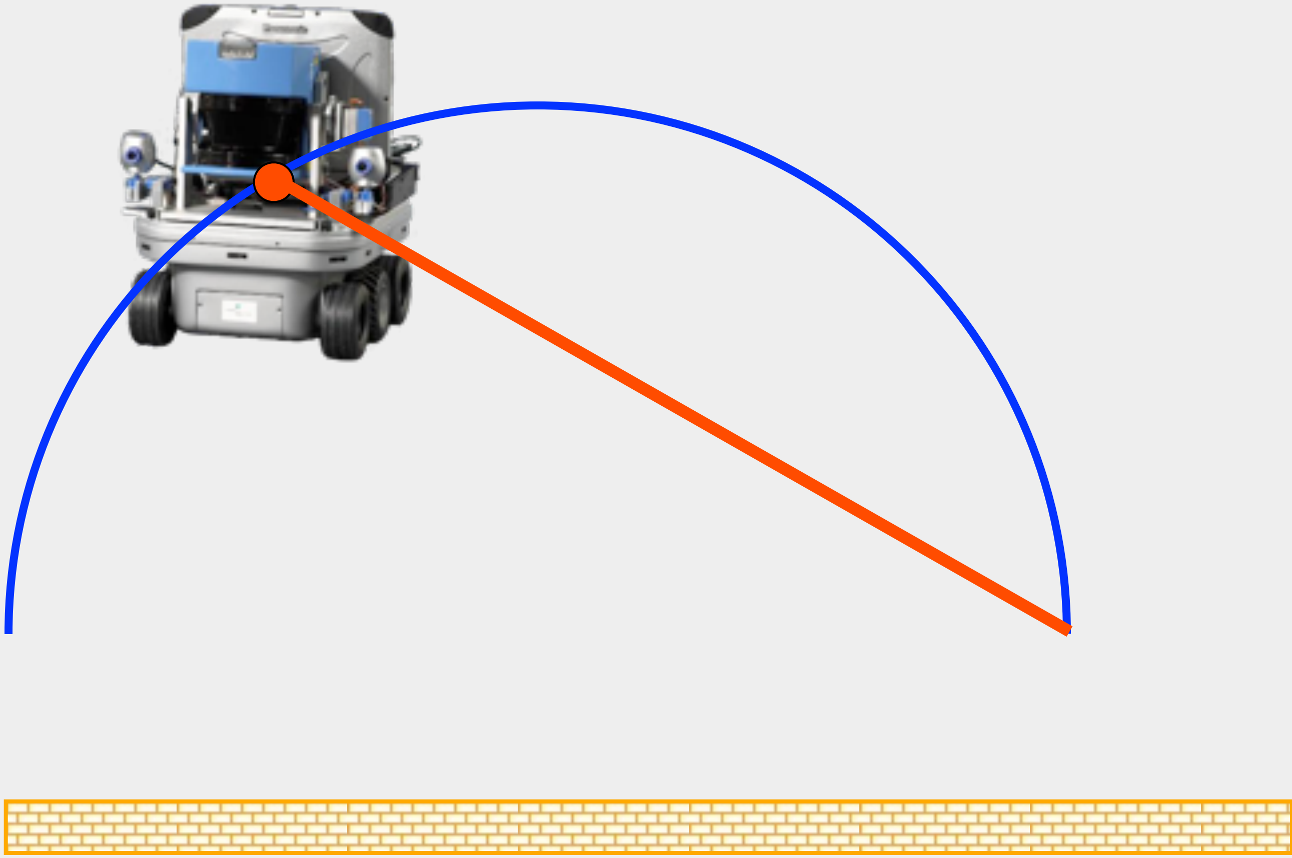
Short Distances

achieved: >4 sc

optimal: 2 sc

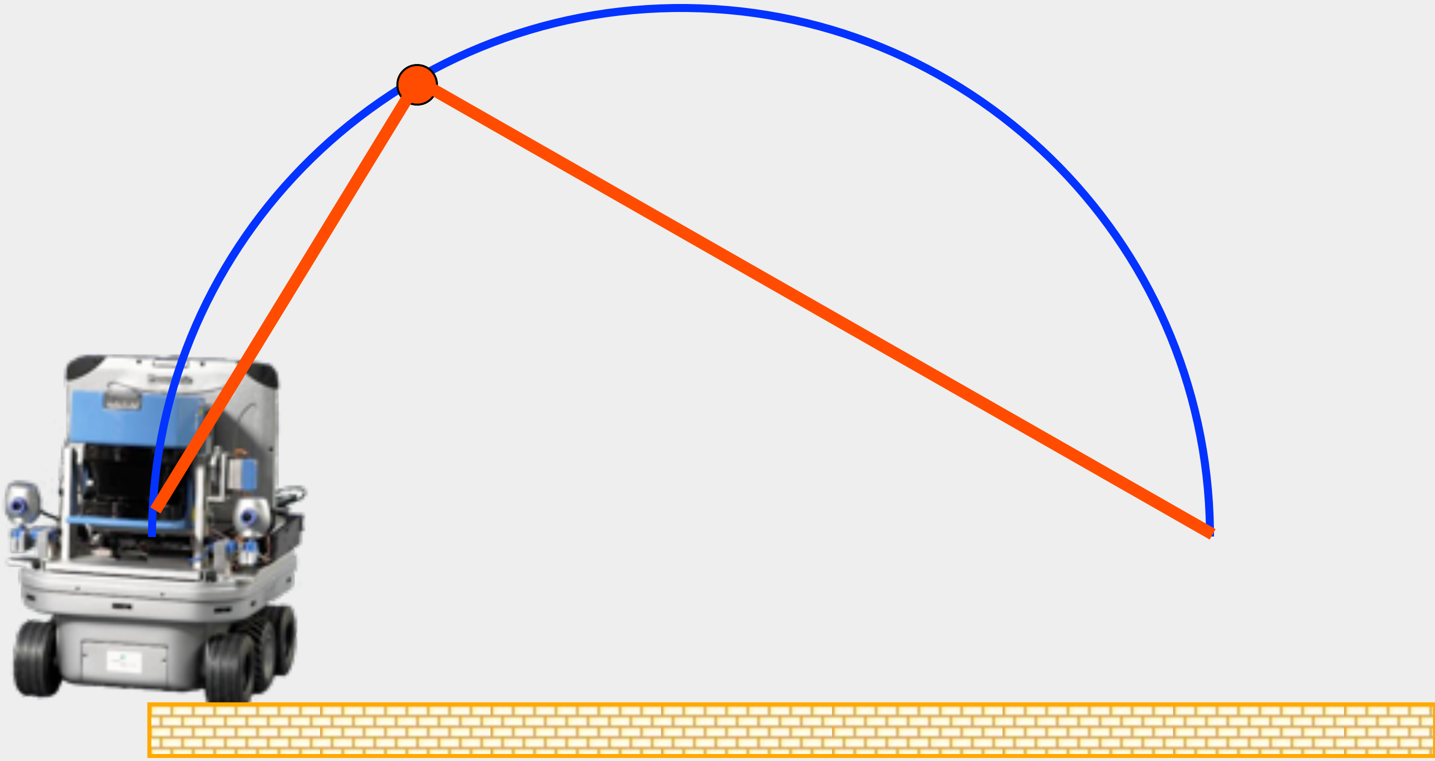


Short Distances



$$\frac{\text{achieved}}{\text{optimal}} = 1.84$$

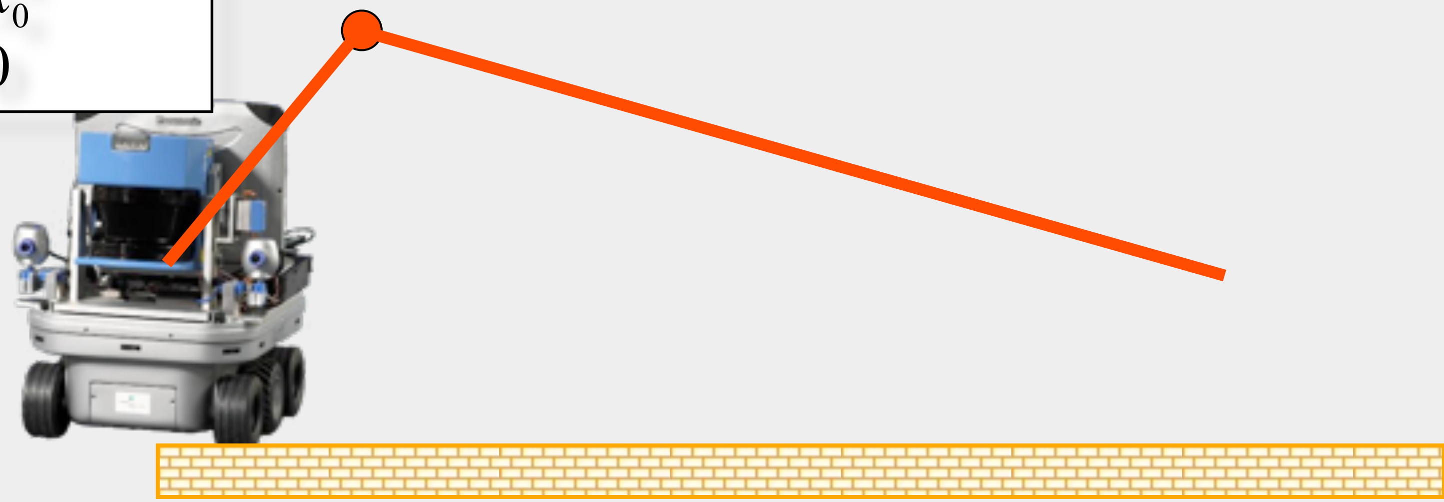
Short Distances



$$\frac{\text{achieved}}{\text{optimal}} = 1.84$$

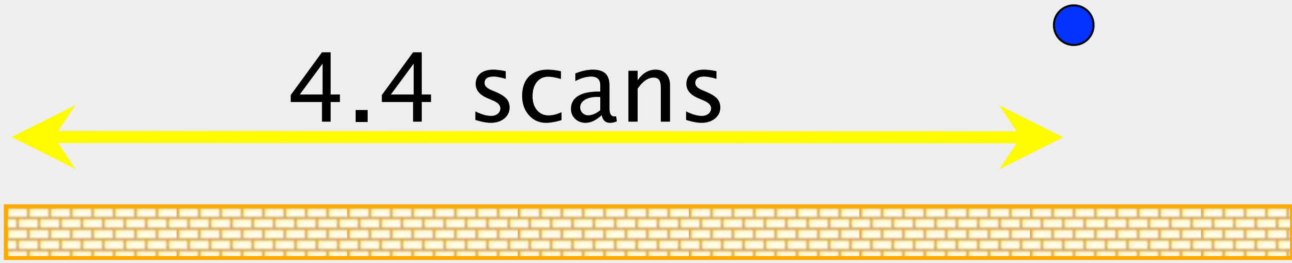
Short Distances

$$\min c$$
$$1 + x_0 = c$$
$$2 + x_0 + x_1 = c(1 + d_0)$$
$$d_0^2 + (y_1 + x_1)^2 = dist^2$$
$$d_0^2 + y_1^2 = x_0^2$$
$$x_0, x_1, d_0, y_1 \geq 0$$



$$\frac{\text{achieved}}{\text{optimal}} = 1.81$$

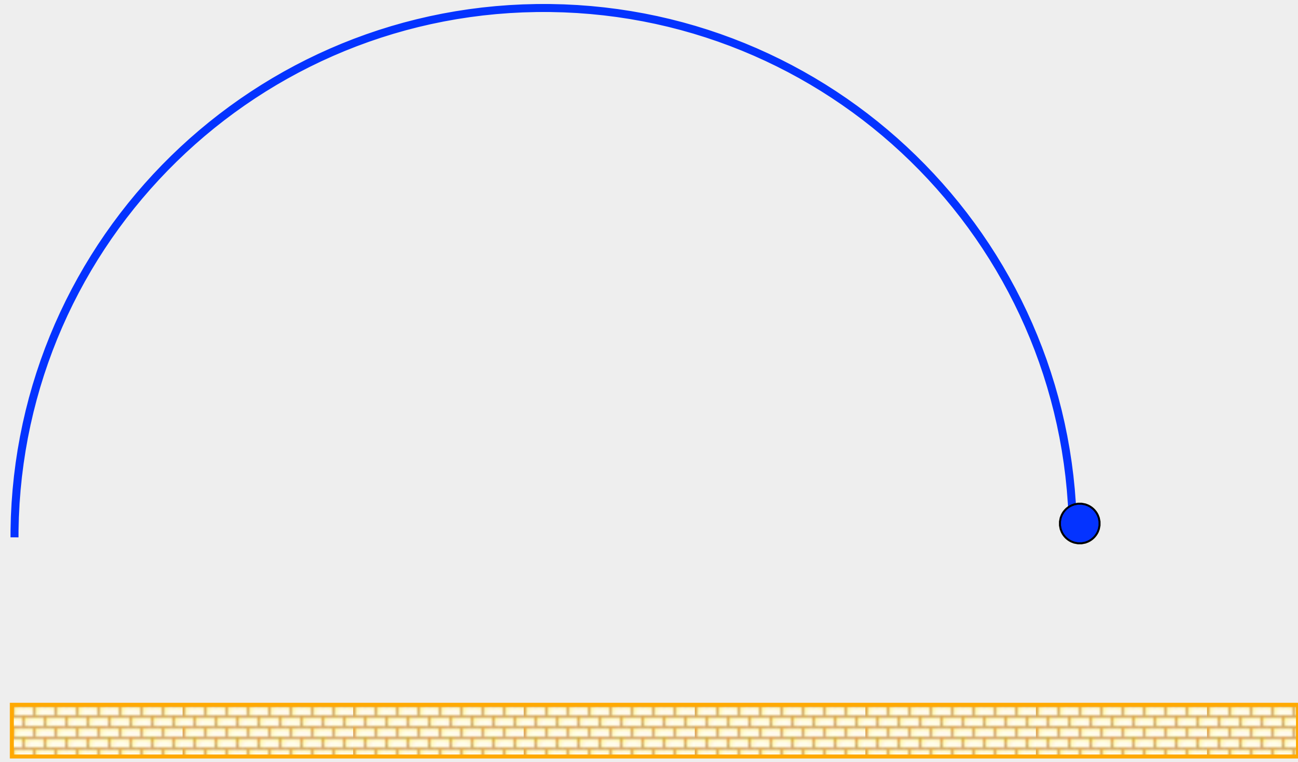
Larger Distances



Larger Distances

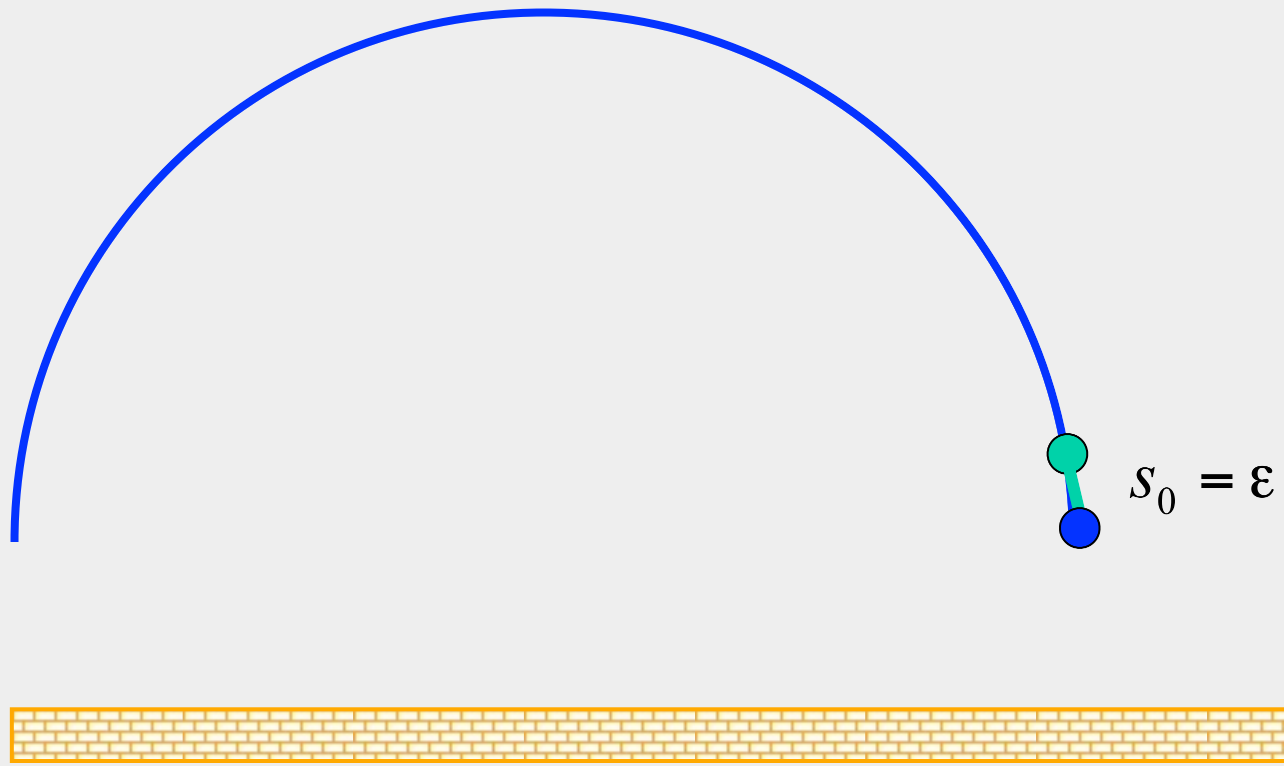
$$\frac{\text{achieved}}{\text{optimal}} = c$$

Larger Distances



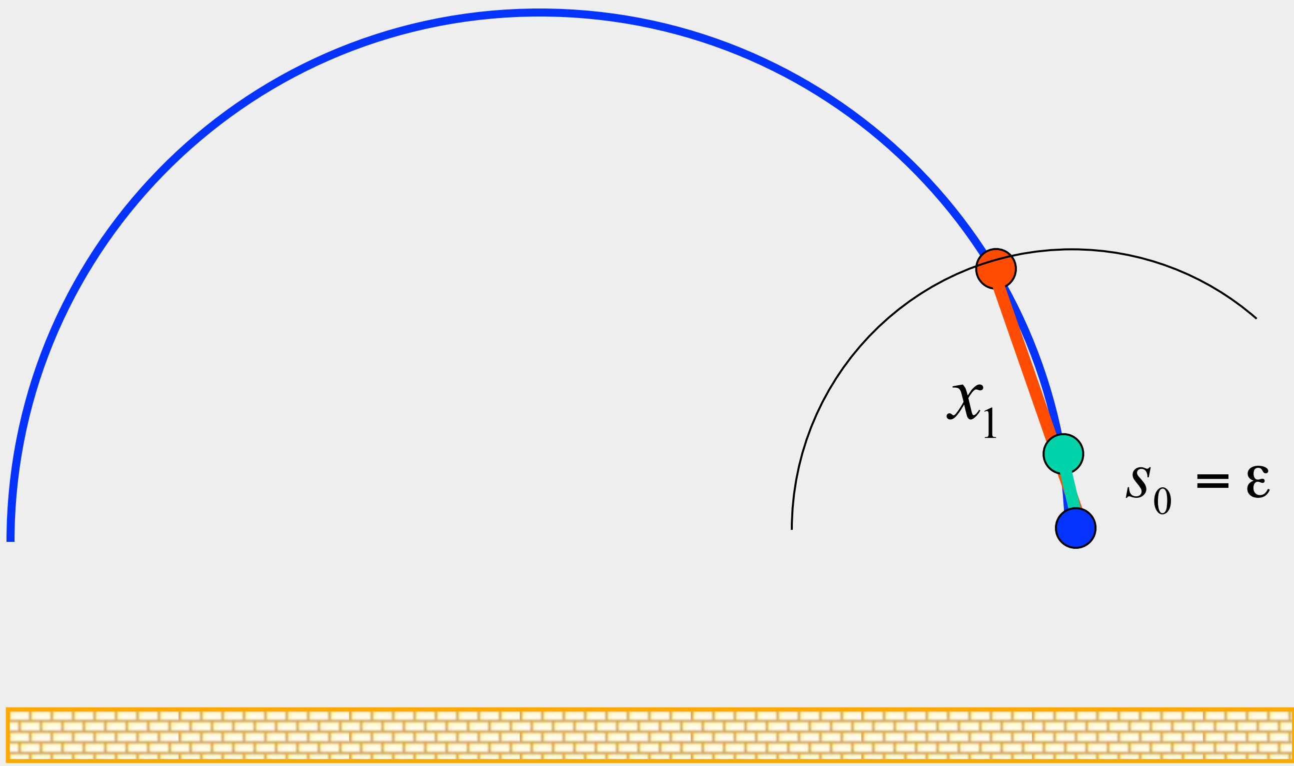
$$\frac{\text{achieved}}{\text{optimal}} = 2.17$$

Larger Distances



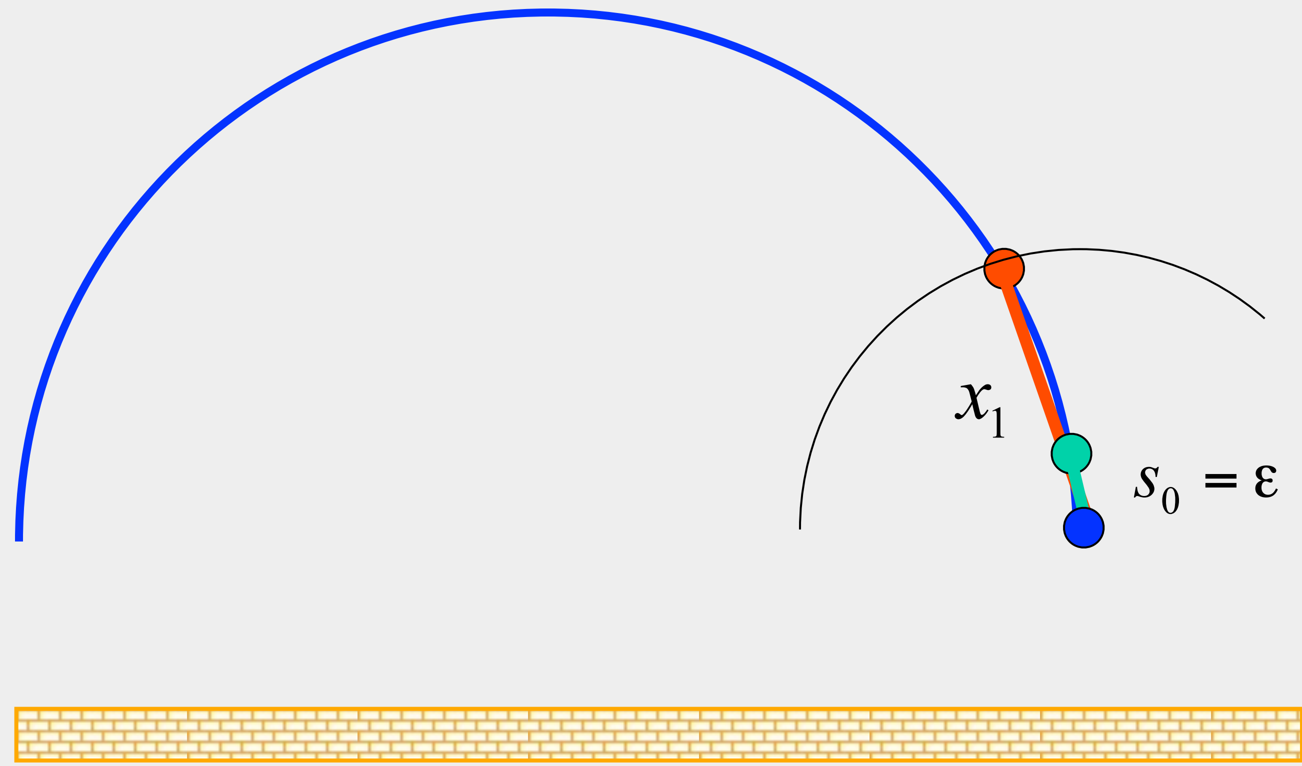
$$\frac{\qquad\qquad}{\varepsilon + 1} = \frac{\text{achieved}}{\text{optimal}} = 2.17$$

Larger Distances



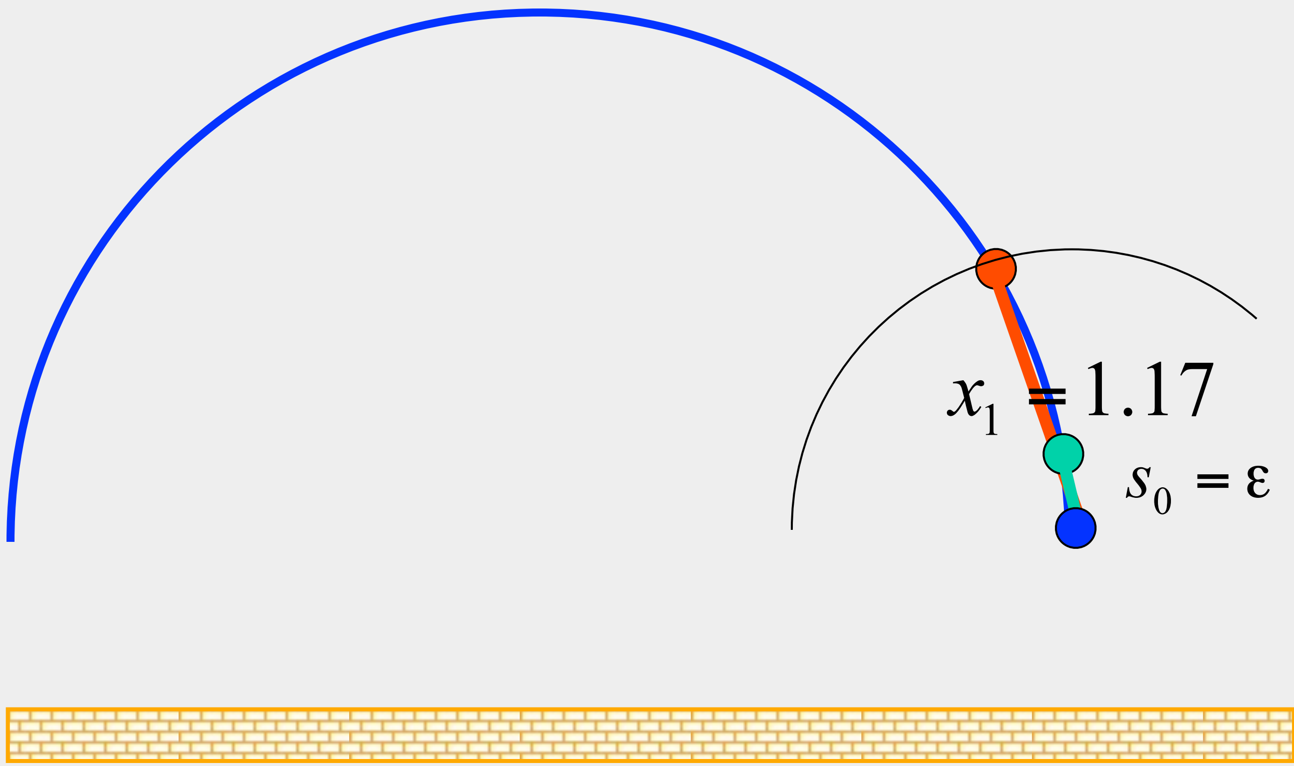
$$\frac{\qquad\qquad}{\varepsilon + 1} = \frac{\text{achieved}}{\text{optimal}} = 2.17$$

Larger Distances



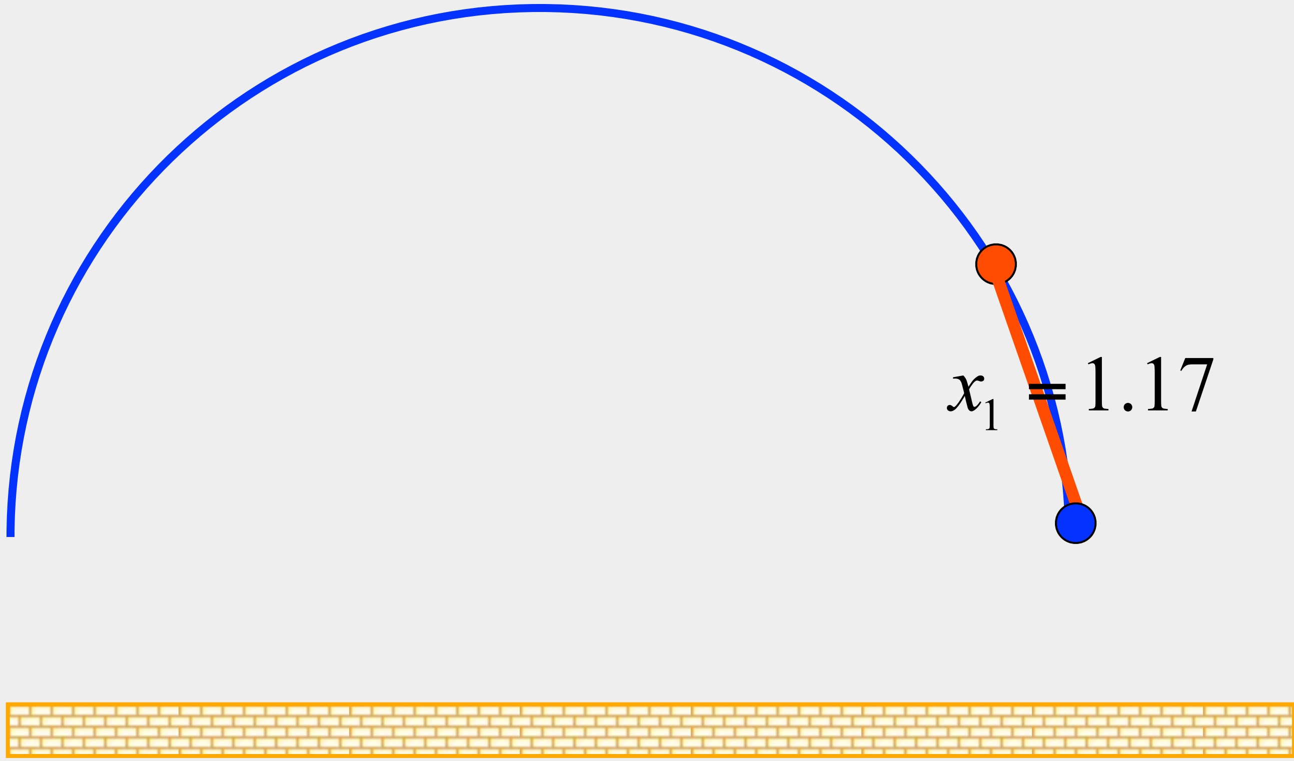
$$\frac{x_1 + 1}{\varepsilon + 1} = \frac{\text{achieved}}{\text{optimal}} = 2.17$$

Larger Distances



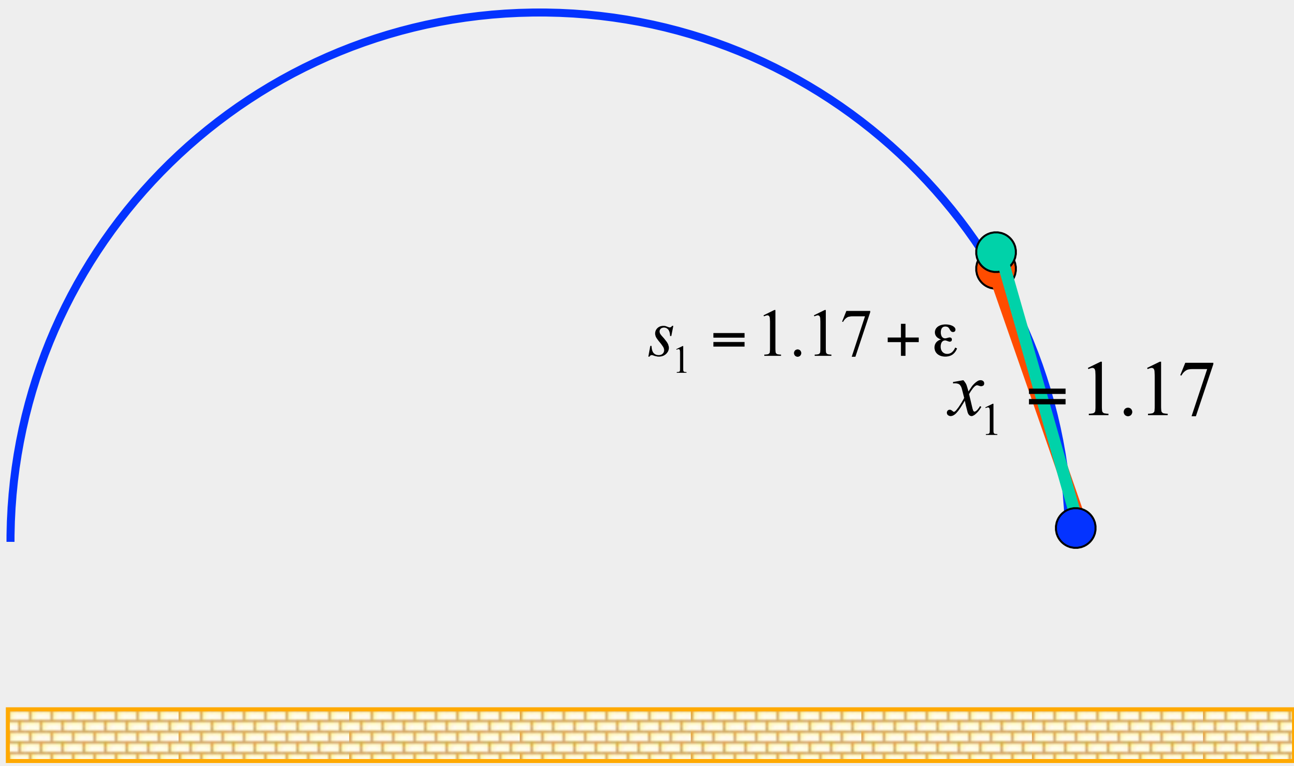
$$\frac{x_1 + 1}{\varepsilon + 1} = \frac{\text{achieved}}{\text{optimal}} = 2.17$$

Larger Distances



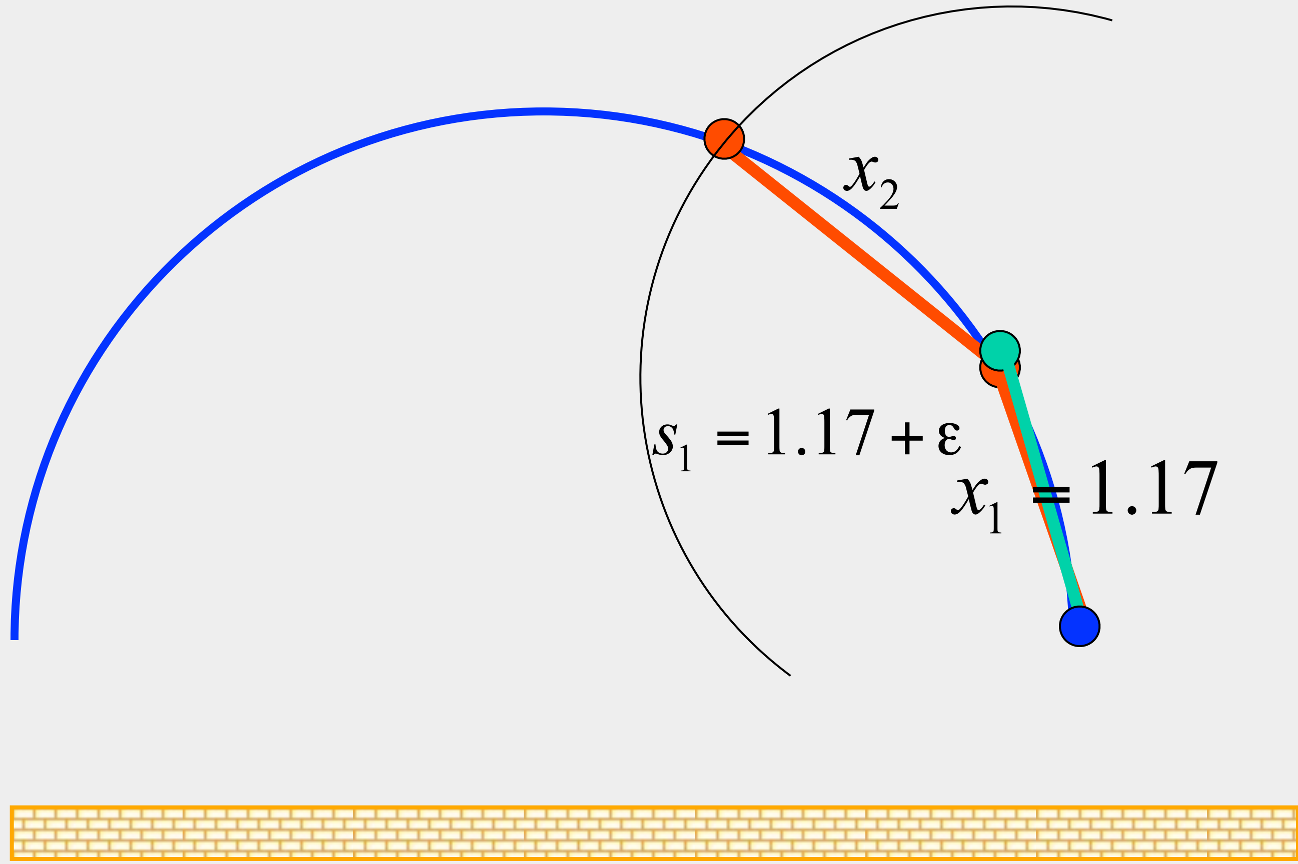
$$\_\_\_\_\_\_\_\_ = \frac{\text{achieved}}{\text{optimal}} = 2.17$$

Larger Distances



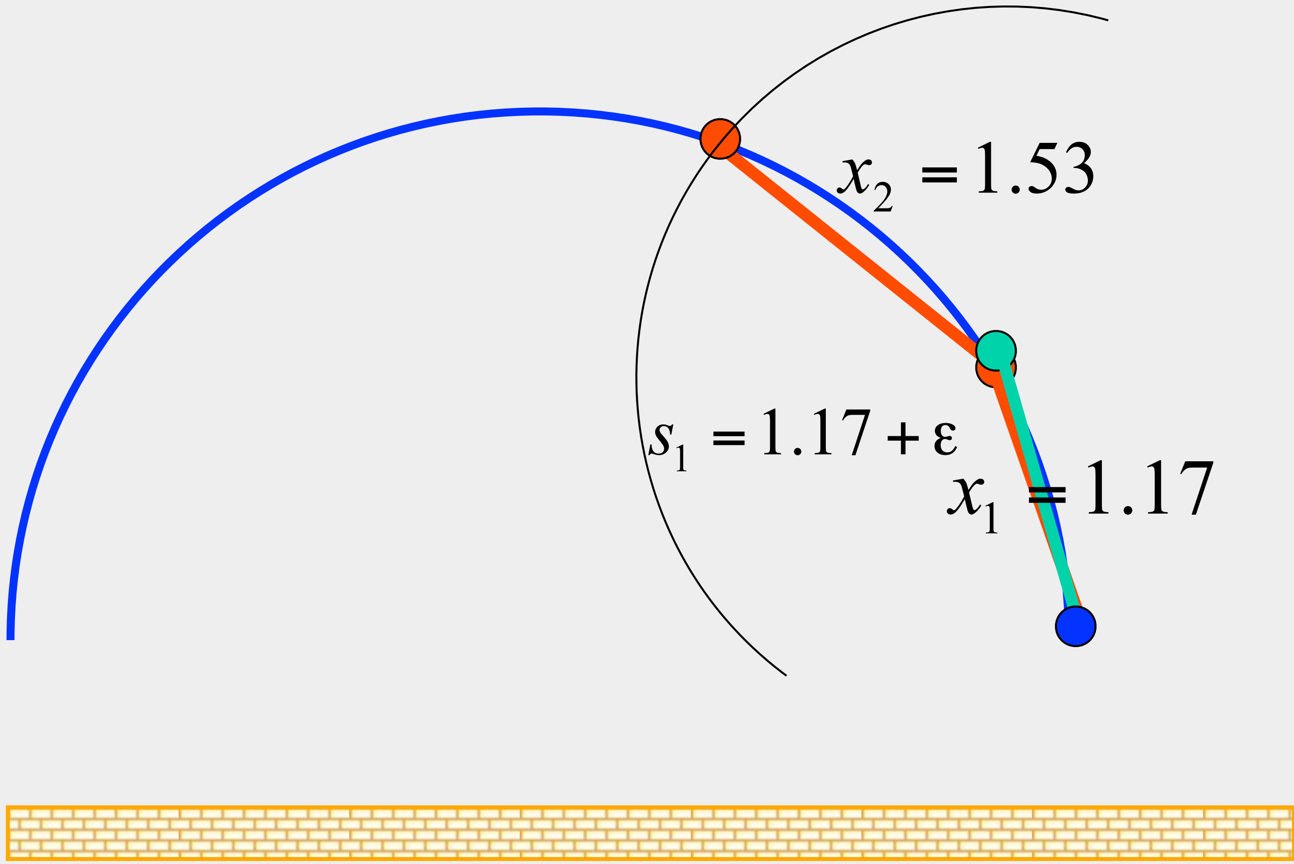
$$\frac{\quad}{1.17 + \varepsilon + 1} = \frac{\text{achieved}}{\text{optimal}} = 2.17$$

Larger Distances



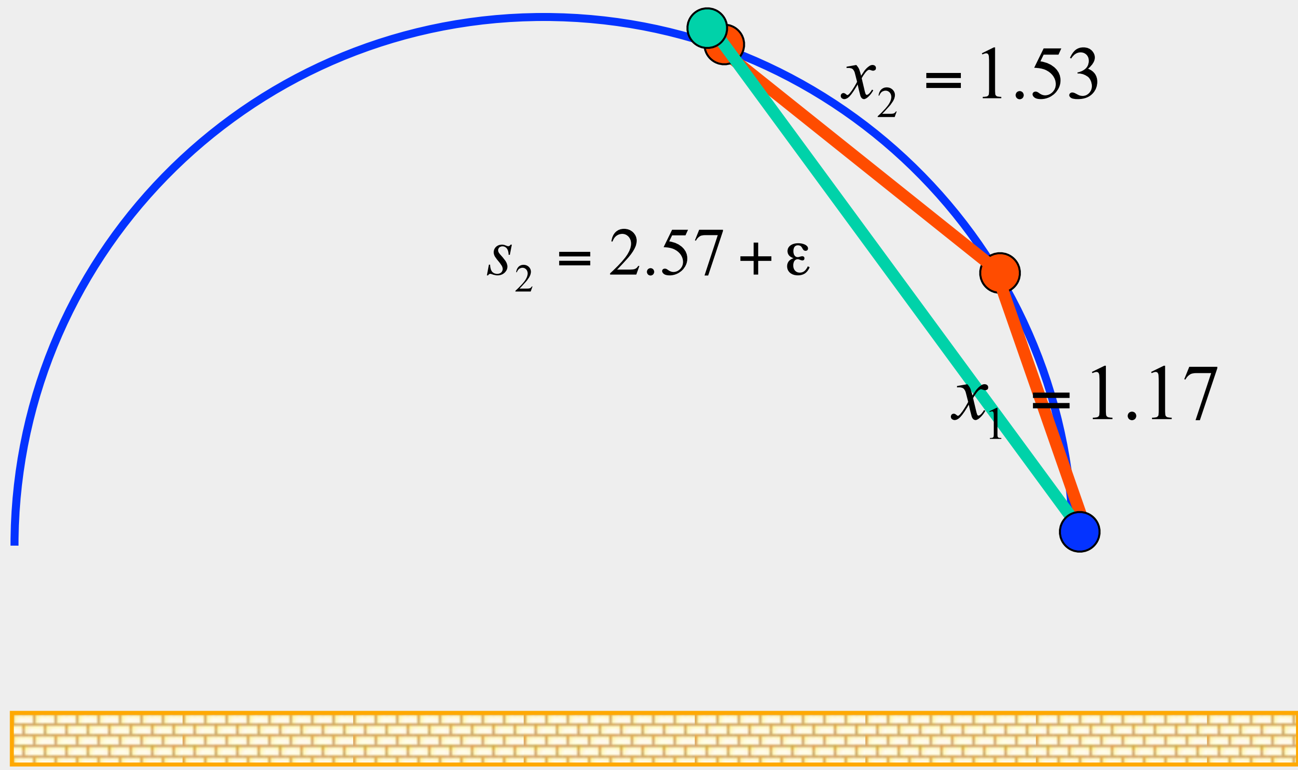
$$\frac{x_2 + 1.17 + 2}{1.17 + \varepsilon + 1} = \frac{\text{achieved}}{\text{optimal}} = 2.17$$

Larger Distances



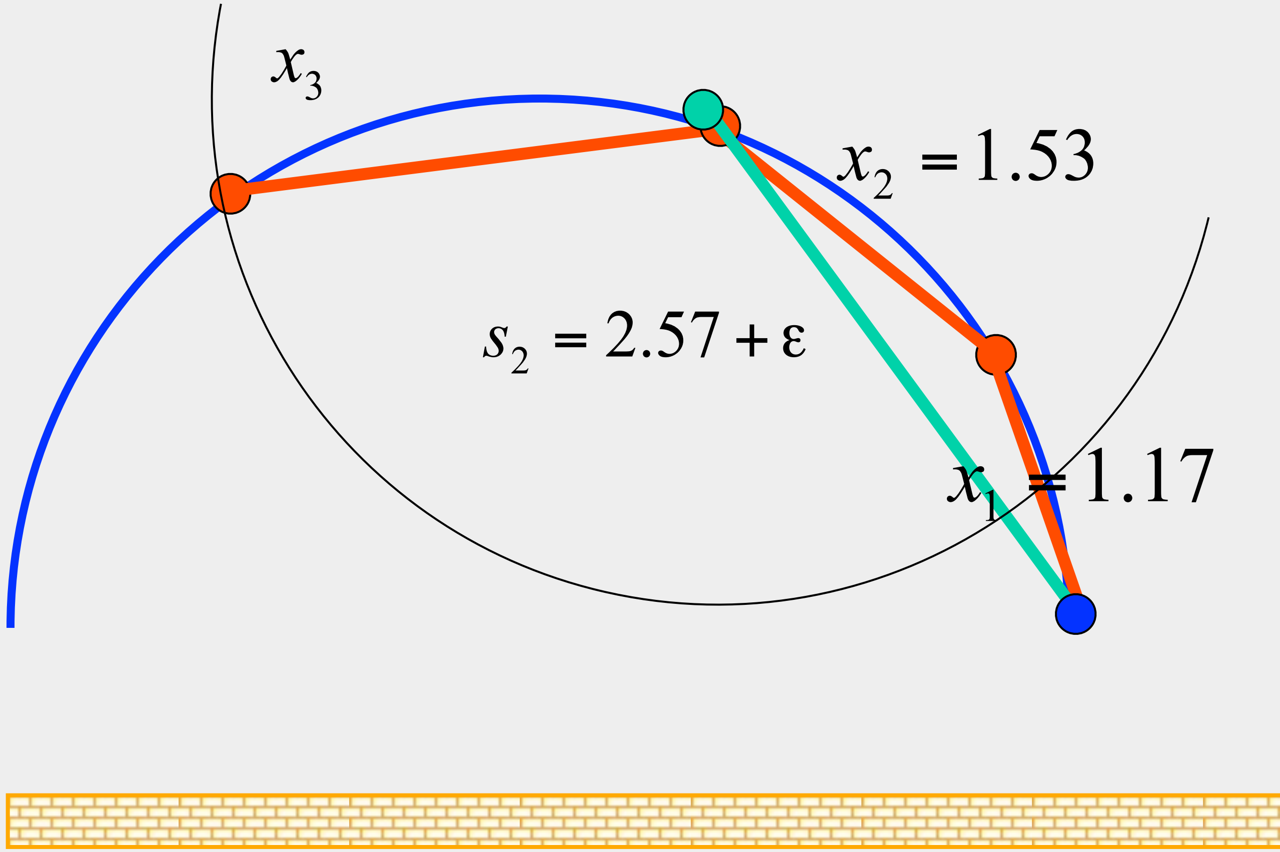
$$\frac{x_2 + 1.17 + 2}{1.17 + \varepsilon + 1} = \frac{\text{achieved}}{\text{optimal}} = 2.17$$

Larger Distances



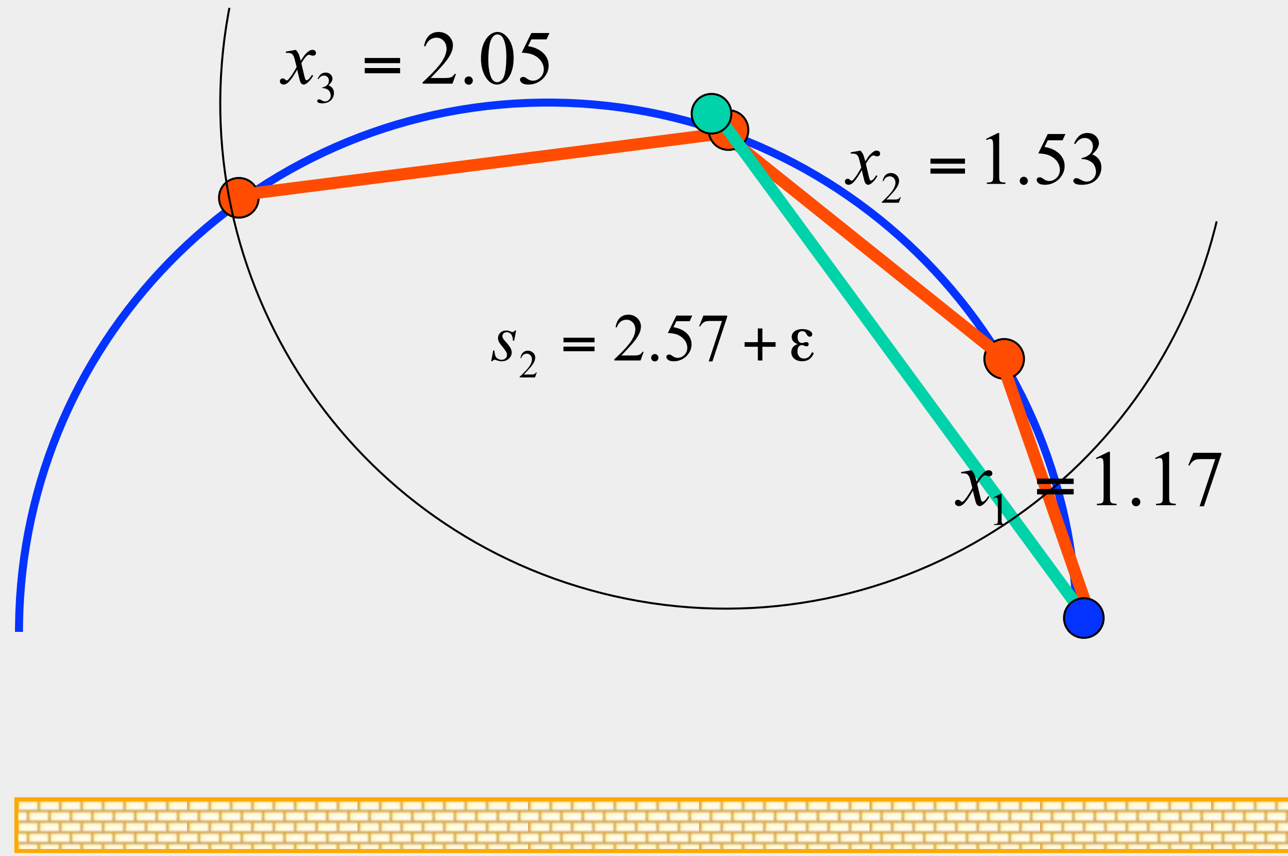
$$\frac{\quad}{2.57 + \varepsilon + 1} = \frac{\text{achieved}}{\text{optimal}} = 2.17$$

Larger Distances



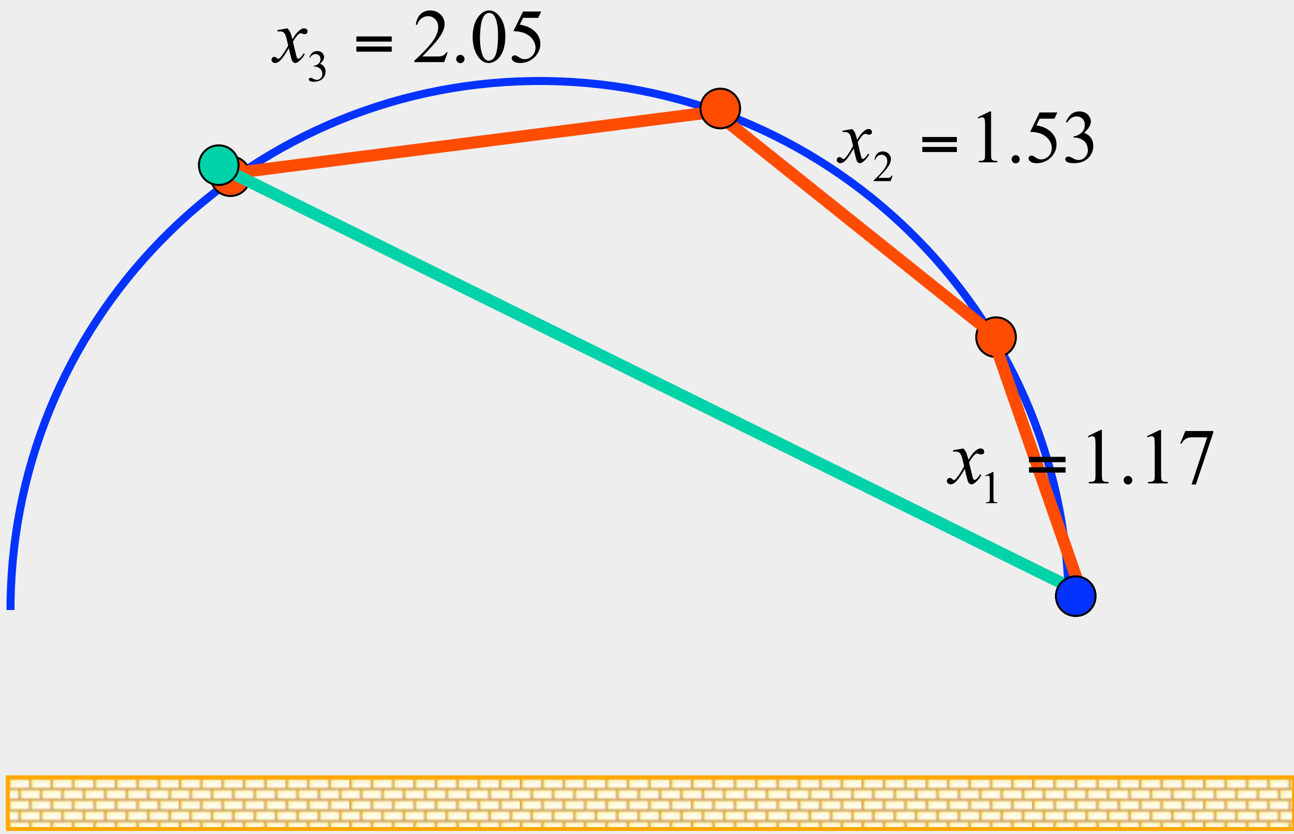
$$\frac{x_3 + 1.53 + 1.17 + 3}{2.57 + \varepsilon + 1} = \frac{\text{achieved}}{\text{optimal}} = 2.17$$

Larger Distances



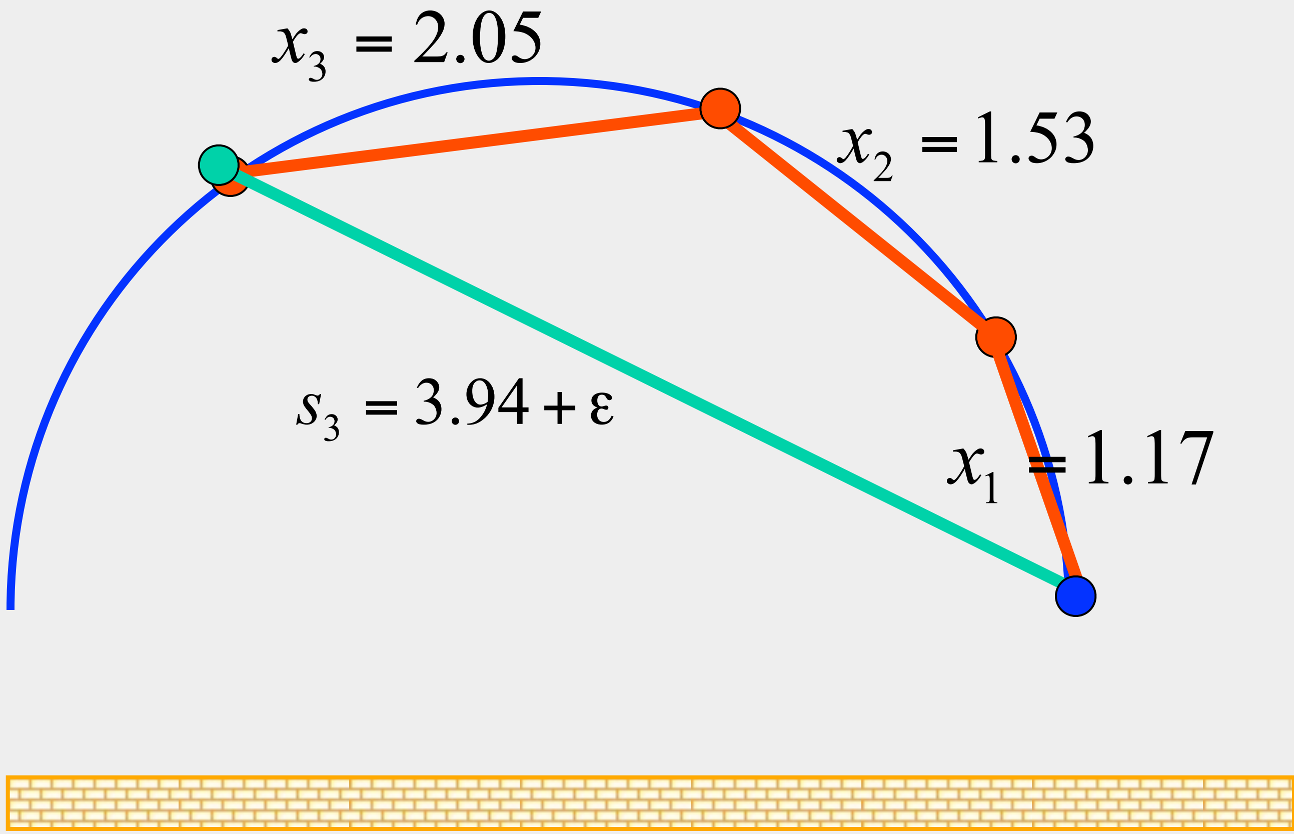
$$\frac{x_3 + 1.53 + 1.17 + 3}{2.57 + \varepsilon + 1} = \frac{\text{achieved}}{\text{optimal}} = 2.17$$

Larger Distances



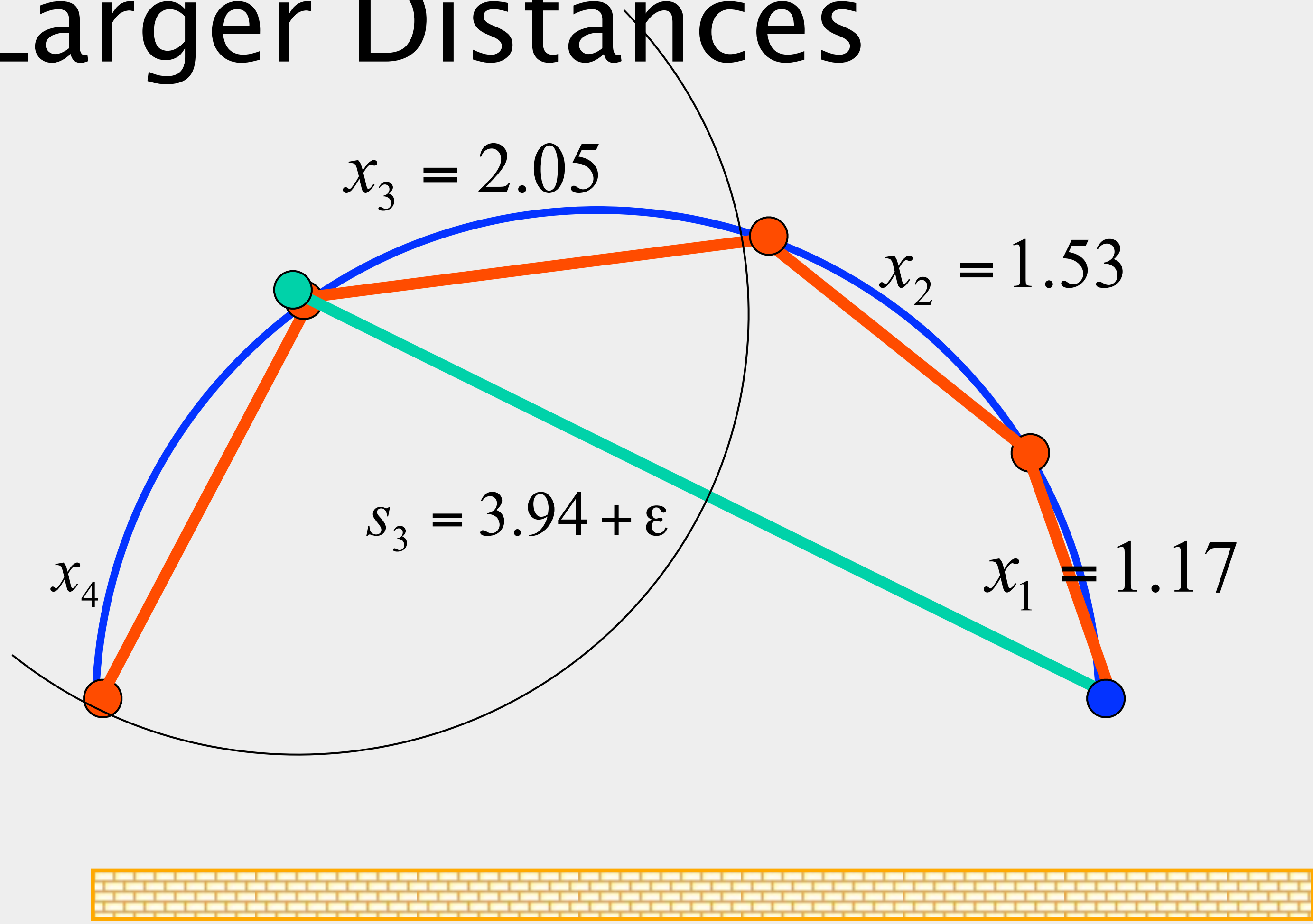
$$\underline{\hspace{6cm}} = \frac{\text{achieved}}{\text{optimal}} = 2.17$$

Larger Distances



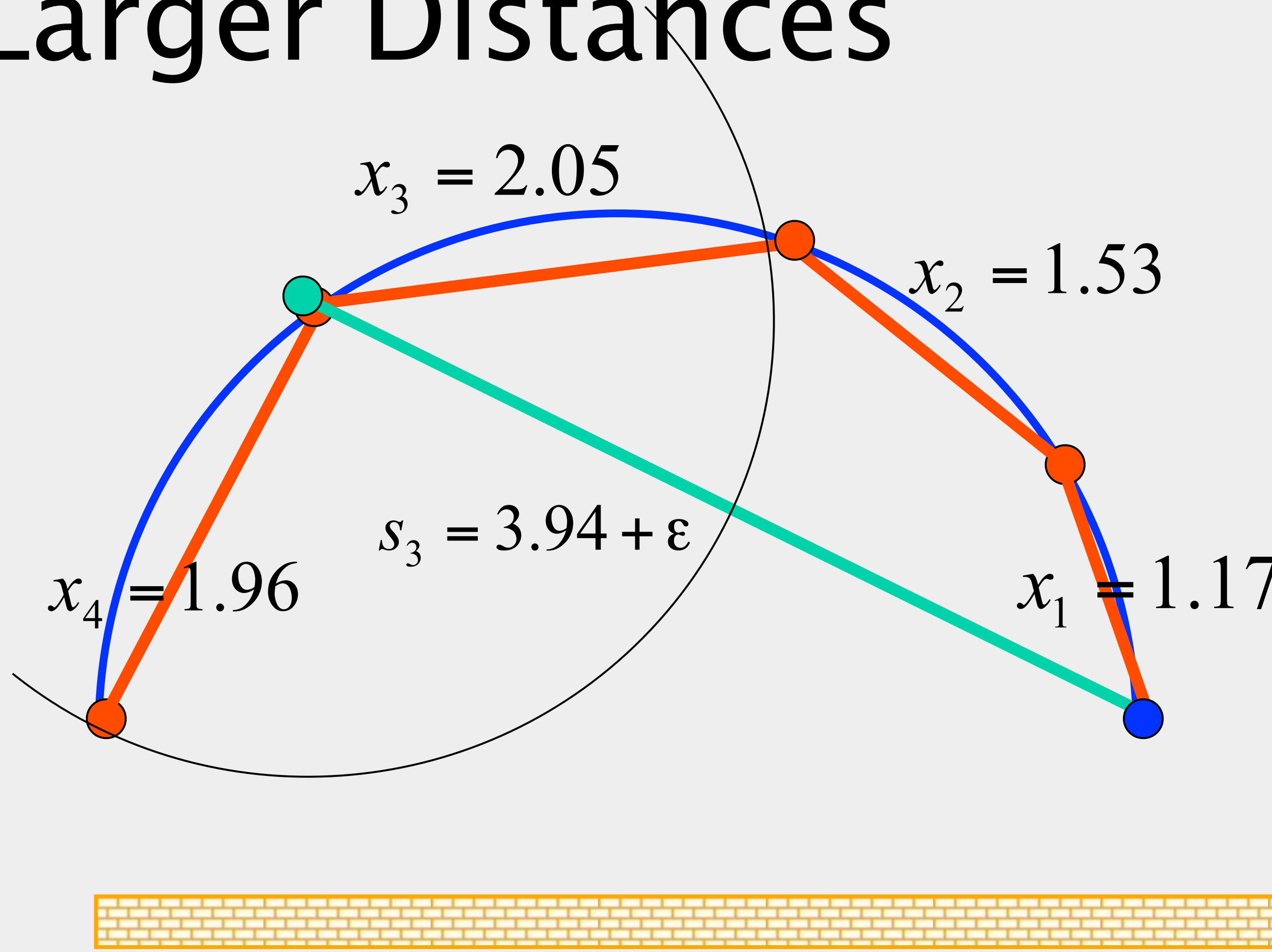
$$\frac{\quad}{3.94 + \varepsilon + 1} = \frac{\text{achieved}}{\text{optimal}} = 2.17$$

Larger Distances



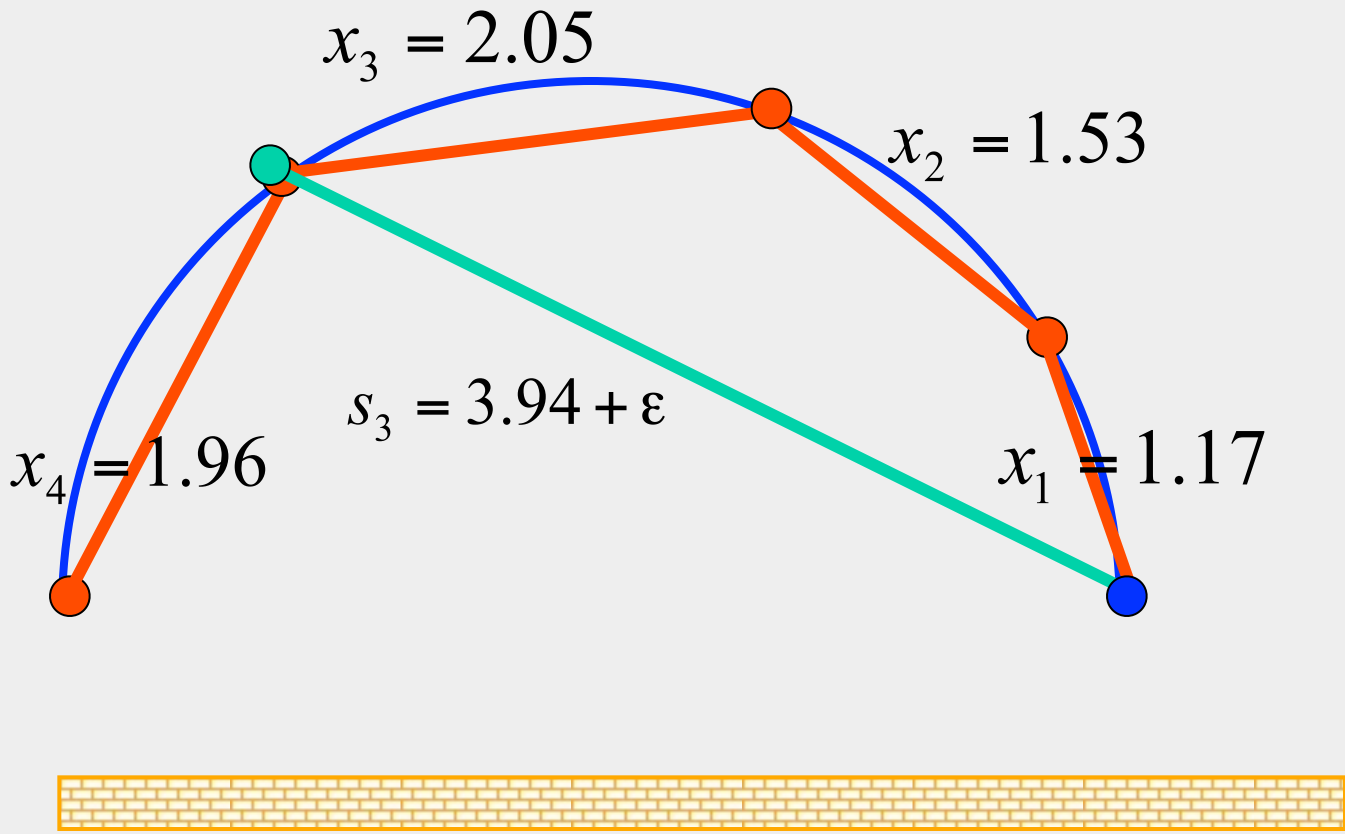
$$\frac{x_4 + 2.05 + 1.53 + 1.17 + 4}{3.94 + \varepsilon + 1} = \frac{\text{achieved}}{\text{optimal}} = 2.17$$

Larger Distances



$$\frac{x_4 + 2.05 + 1.53 + 1.17 + 4}{3.94 + \varepsilon + 1} = \frac{\text{achieved}}{\text{optimal}} = 2.17$$

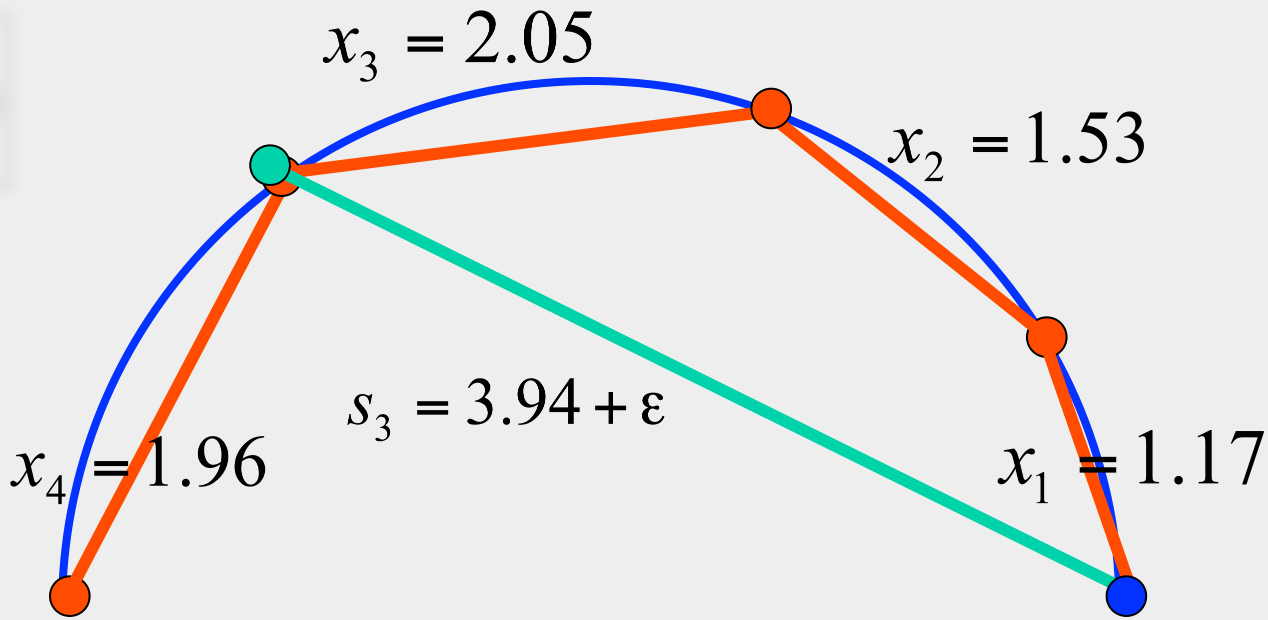
Larger Distances



$$\frac{x_4 + 2.05 + 1.53 + 1.17 + 4}{3.94 + \varepsilon + 1} = \frac{\text{achieved}}{\text{optimal}} = 2.17$$

Larger Distances

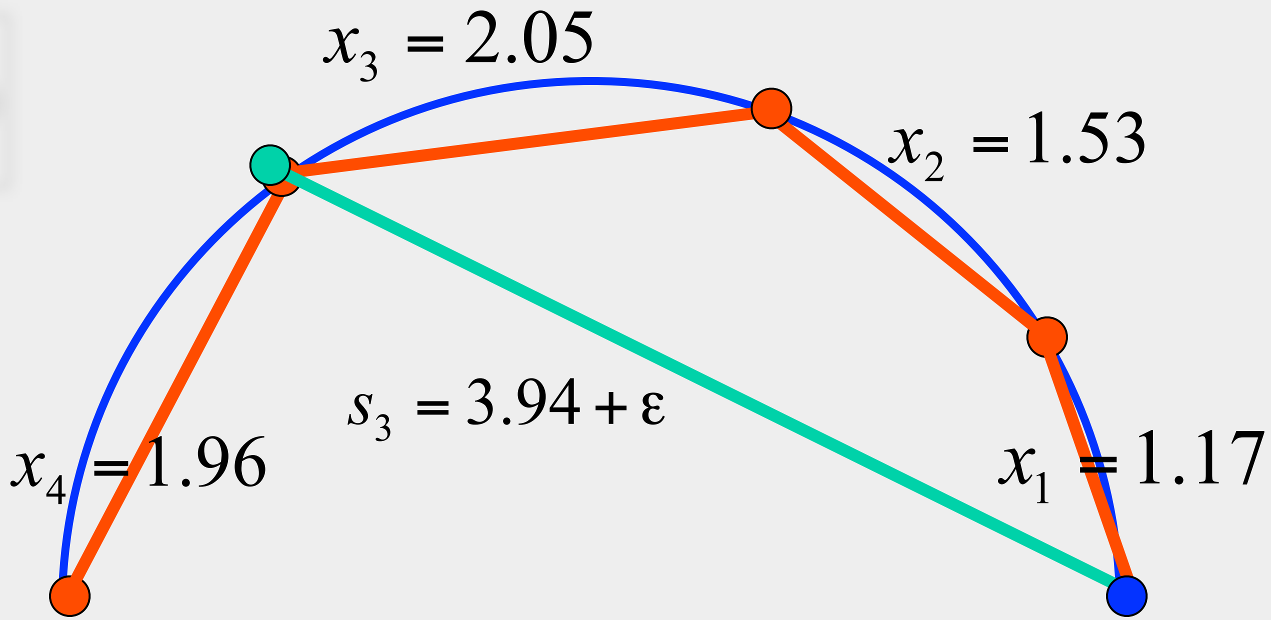
$$n + \sum_{i=1}^{n} x_i = c(s_{n-1} + 1)$$



$$\frac{x_4 + 2.05 + 1.53 + 1.17 + 4}{3.94 + \varepsilon + 1} = \frac{\text{achieved}}{\text{optimal}} = 2.17$$

Larger Distances

$$n + \sum_{i=1}^{n} x_i = c(s_{n-1} + 1)$$

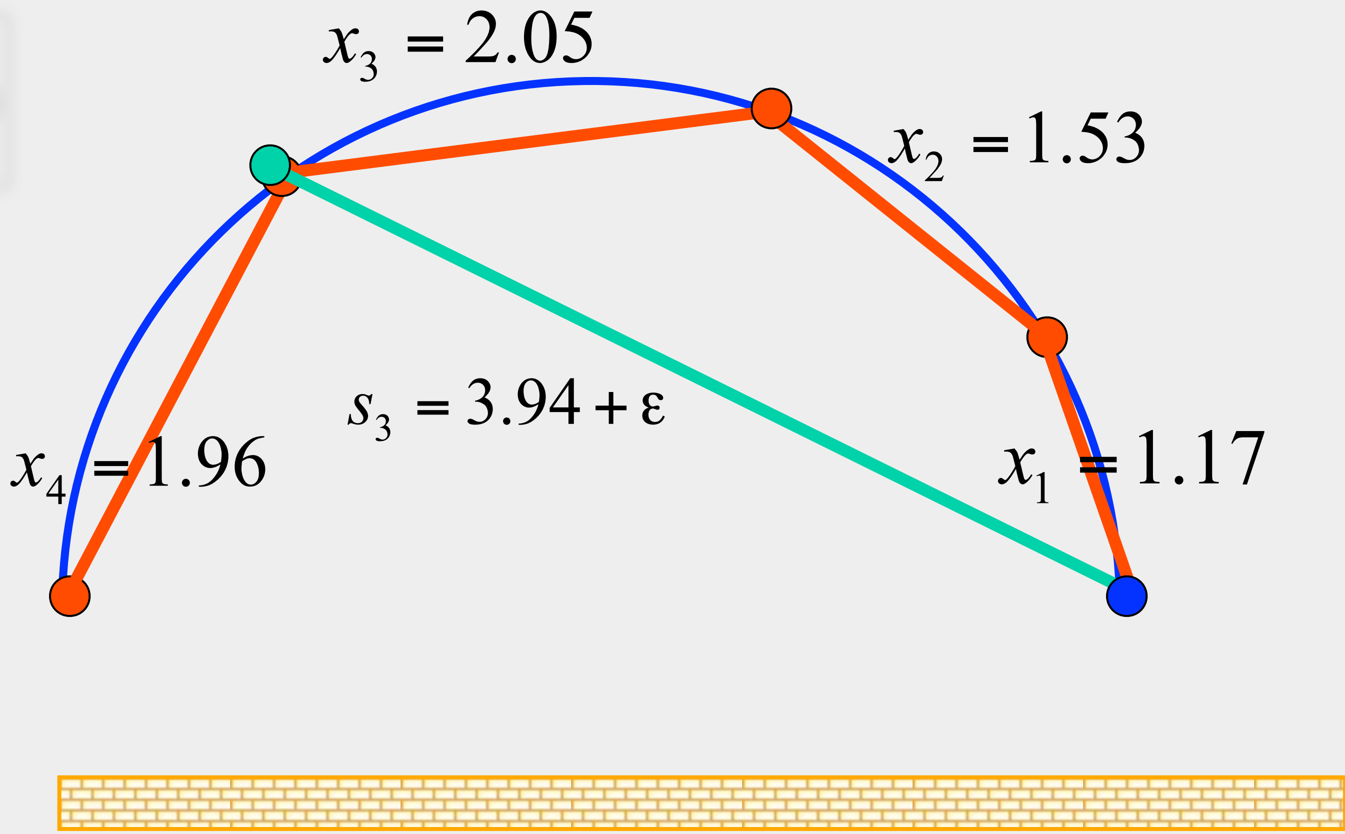


$$\frac{\text{achieved}}{\text{optimal}} = 2.17$$

Larger Distances

$$n + \sum_{i=1}^{n} x_i = c(s_{n-1} + 1)$$

$$s_{n-1} = d \sin w_{n-1}$$

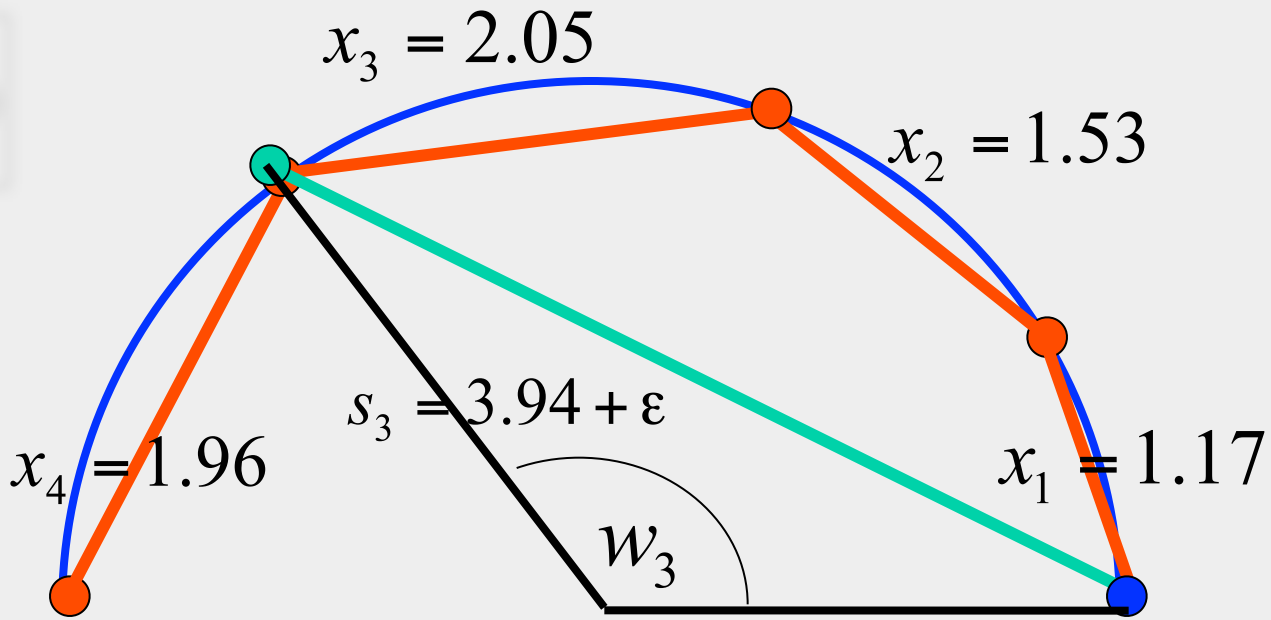


$$\frac{\text{achieved}}{\text{optimal}} = 2.17$$

Larger Distances

$$n + \sum_{i=1}^{n} x_i = c(s_{n-1} + 1)$$

$$s_{n-1} = d \sin w_{n-1}$$



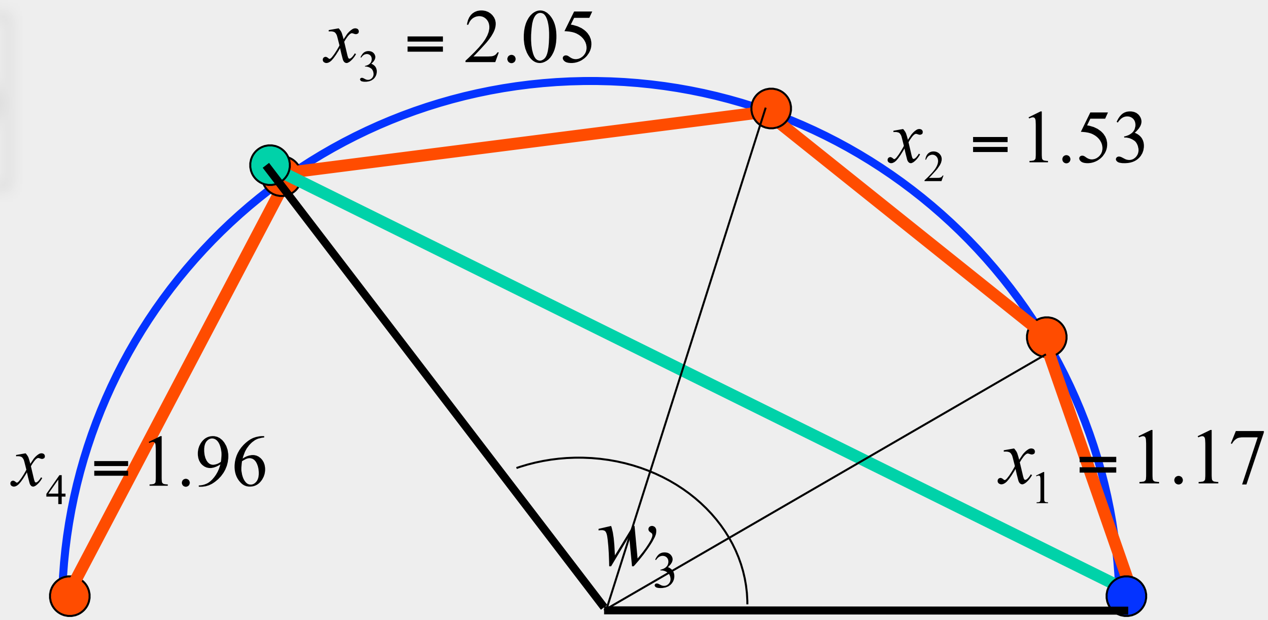
$$\frac{\text{achieved}}{\text{optimal}} = 2.17$$

Larger Distances

$$n + \sum_{i=1}^{n} x_i = c(s_{n-1} + 1)$$

$$s_{n-1} = d \sin w_{n-1}$$

$$w_{n-1} = \sum_{i=1}^{n-1} \varphi_i$$



$$\frac{\text{achieved}}{\text{optimal}} = 2.17$$

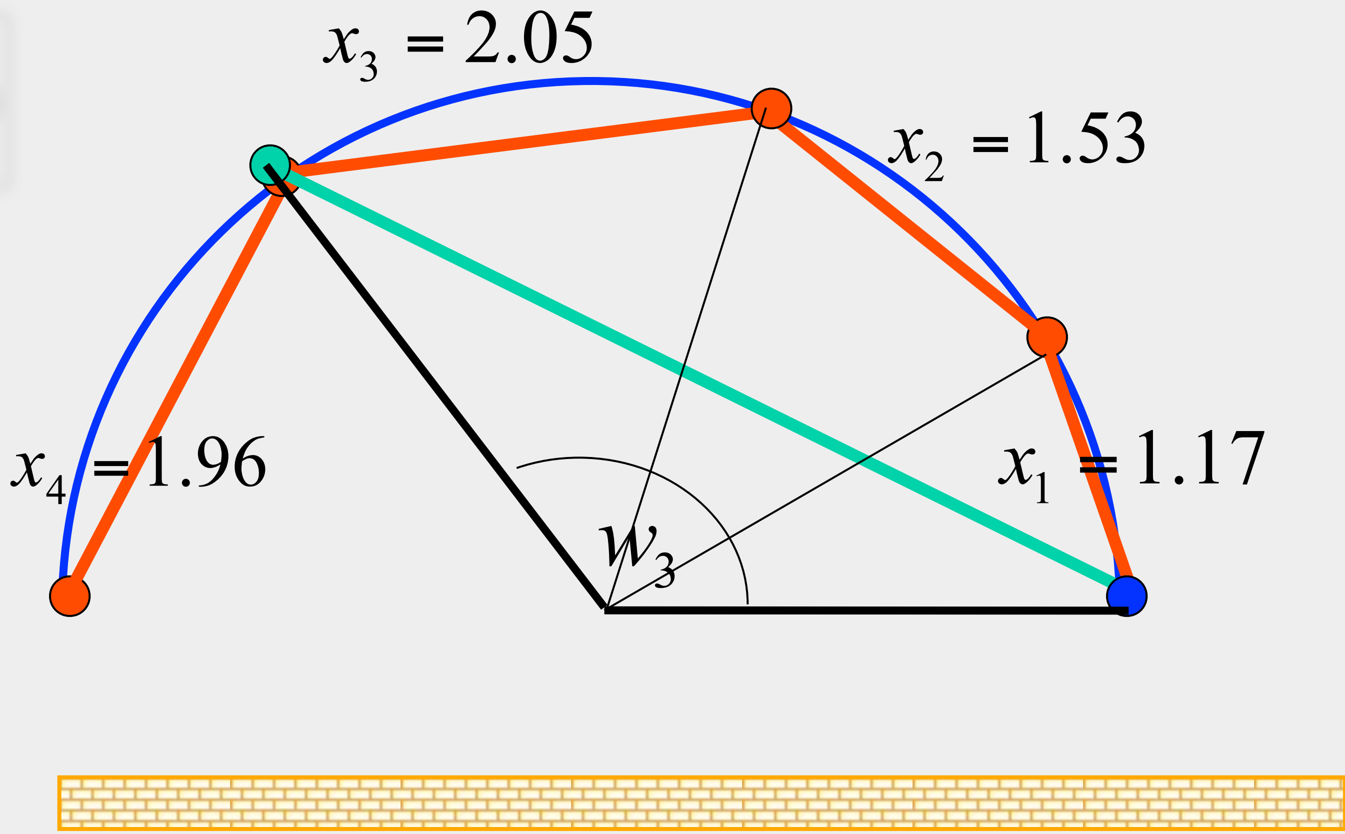
Larger Distances

$$n + \sum_{i=1}^{n} x_i = c(s_{n-1} + 1)$$

$$s_{n-1} = d \sin w_{n-1}$$

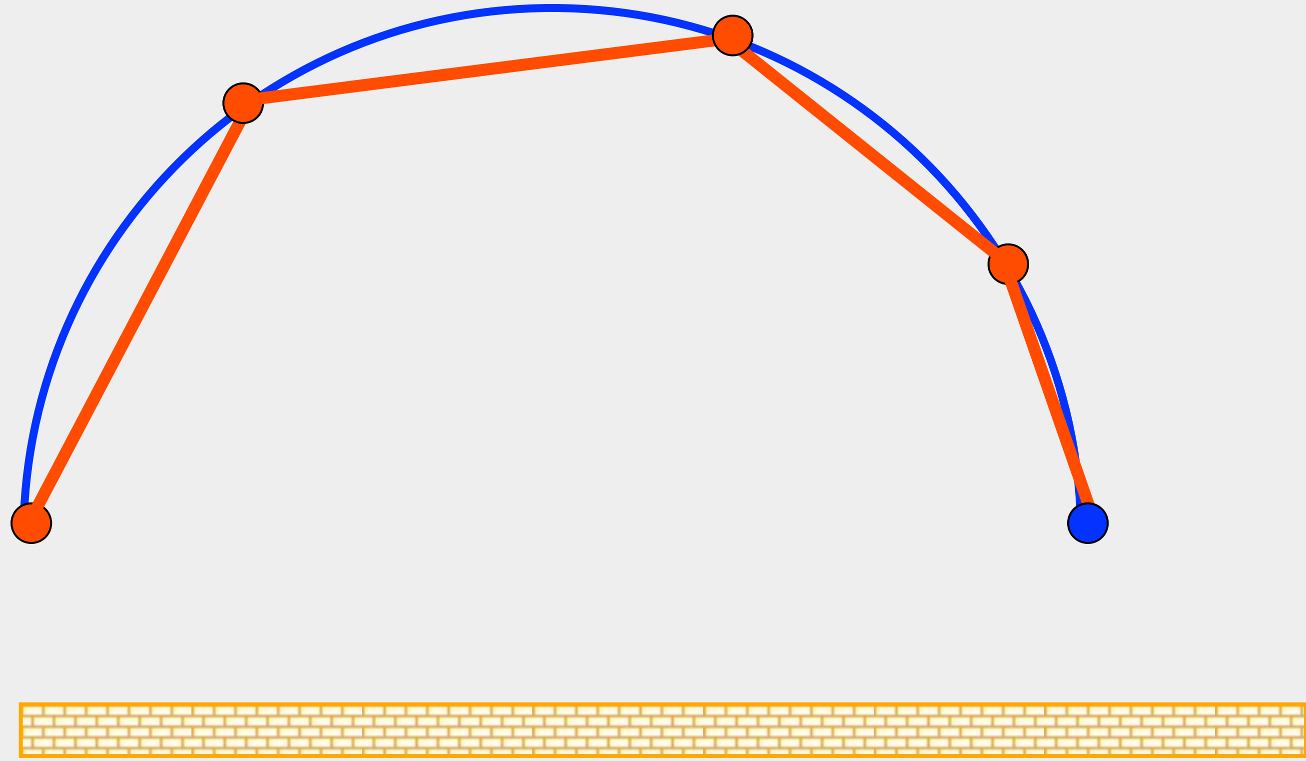
$$w_{n-1} = \sum_{i=1}^{n-1} \varphi_i$$

$$\varphi_i = 2 \arcsin\left(\frac{x_i}{d}\right)$$



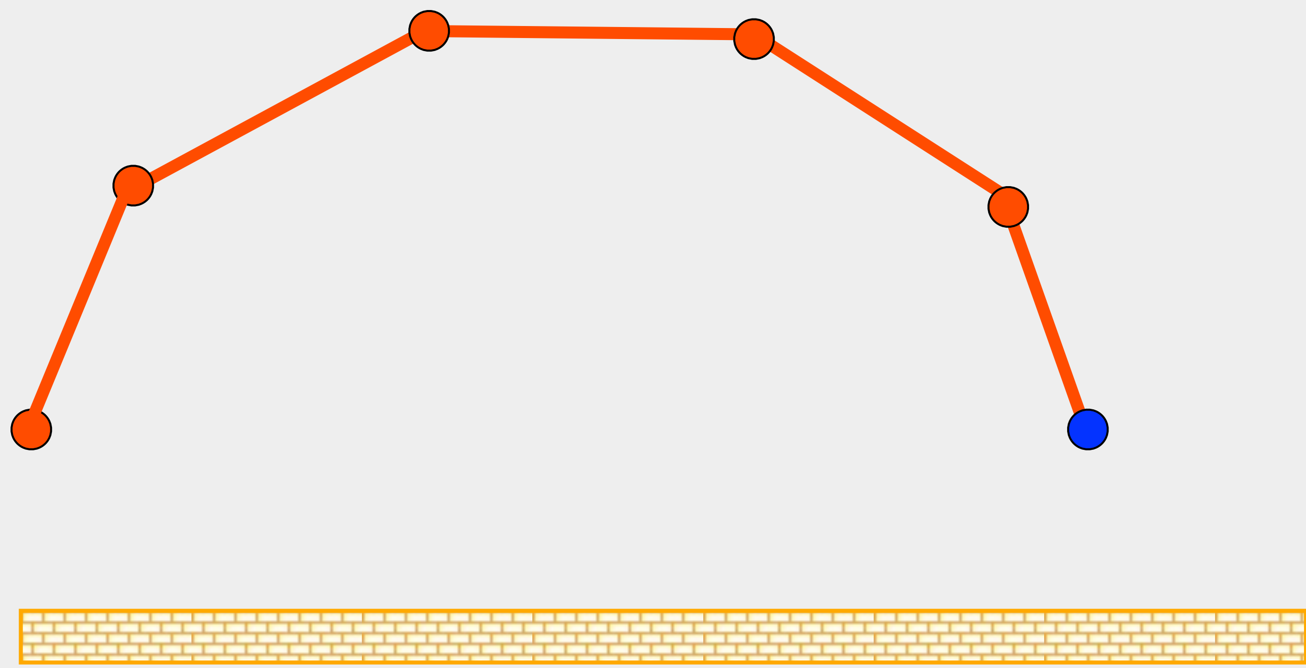
$$\frac{\text{achieved}}{\text{optimal}} = 2.17$$

Larger Distances



$$\frac{\text{achieved}}{\text{optimal}} = 2.17$$

Larger Distances



$$\frac{\text{achieved}}{\text{optimal}} = 2.12$$

A Lower Bound

Theorem: No strategy can guarantee a competitive factor below 2.

Sketch: Assume factor $c = 2 - \delta$.

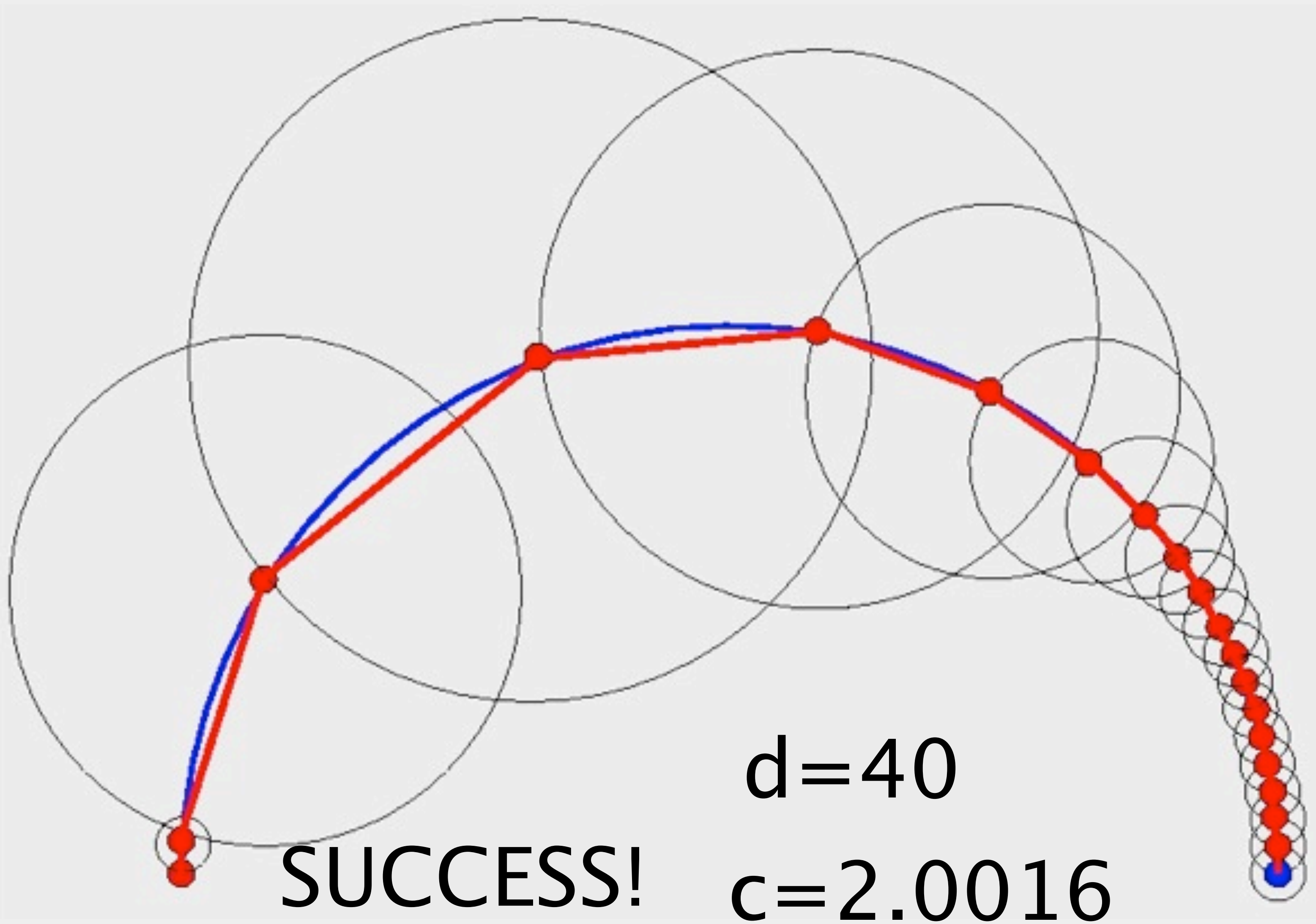
Use induction to show that for the i-th step length, we have $x_i \leq (1-\delta)^i$.

Then the total distance is bounded by

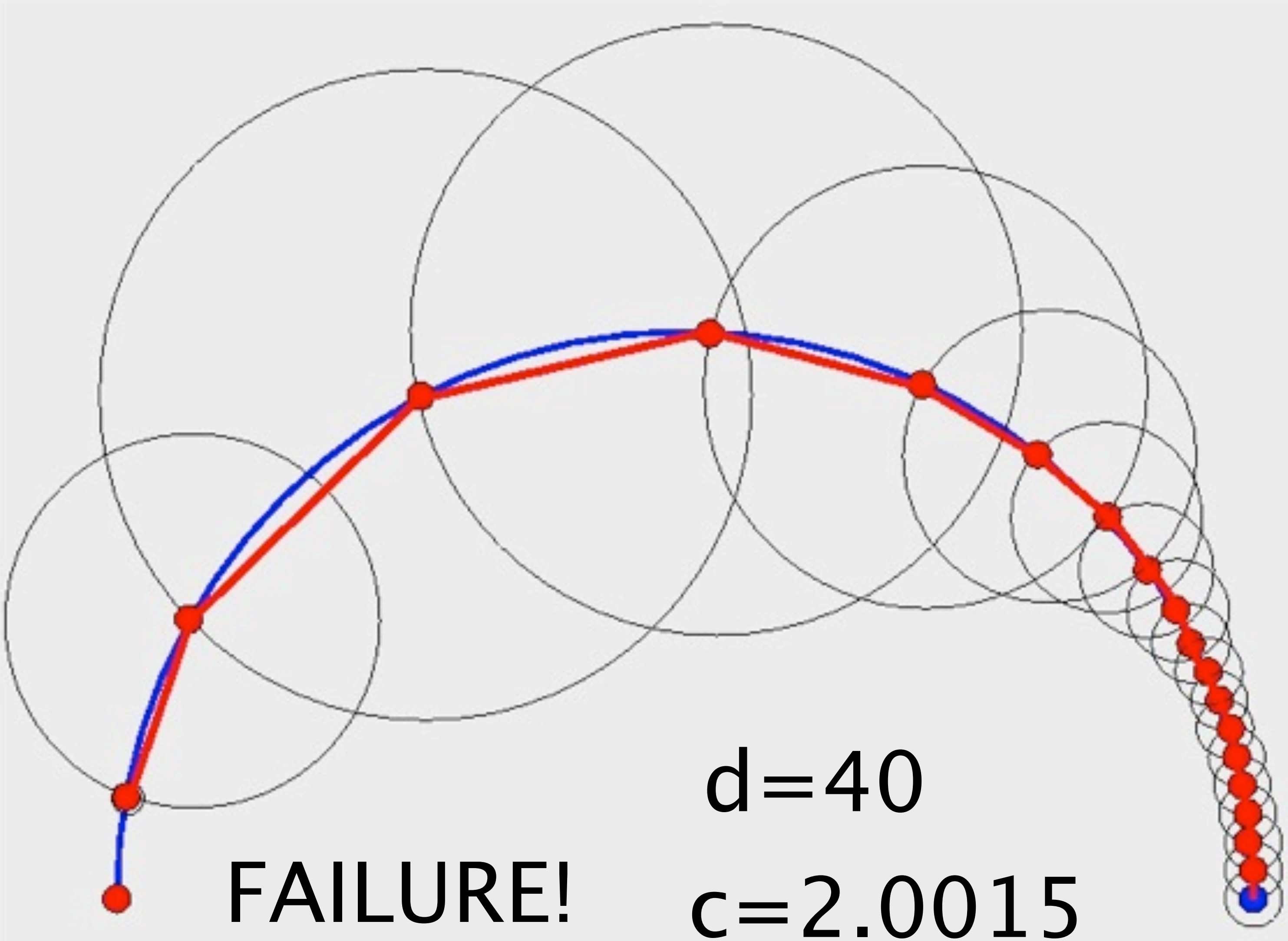
$$1/\delta,$$

a contradiction.

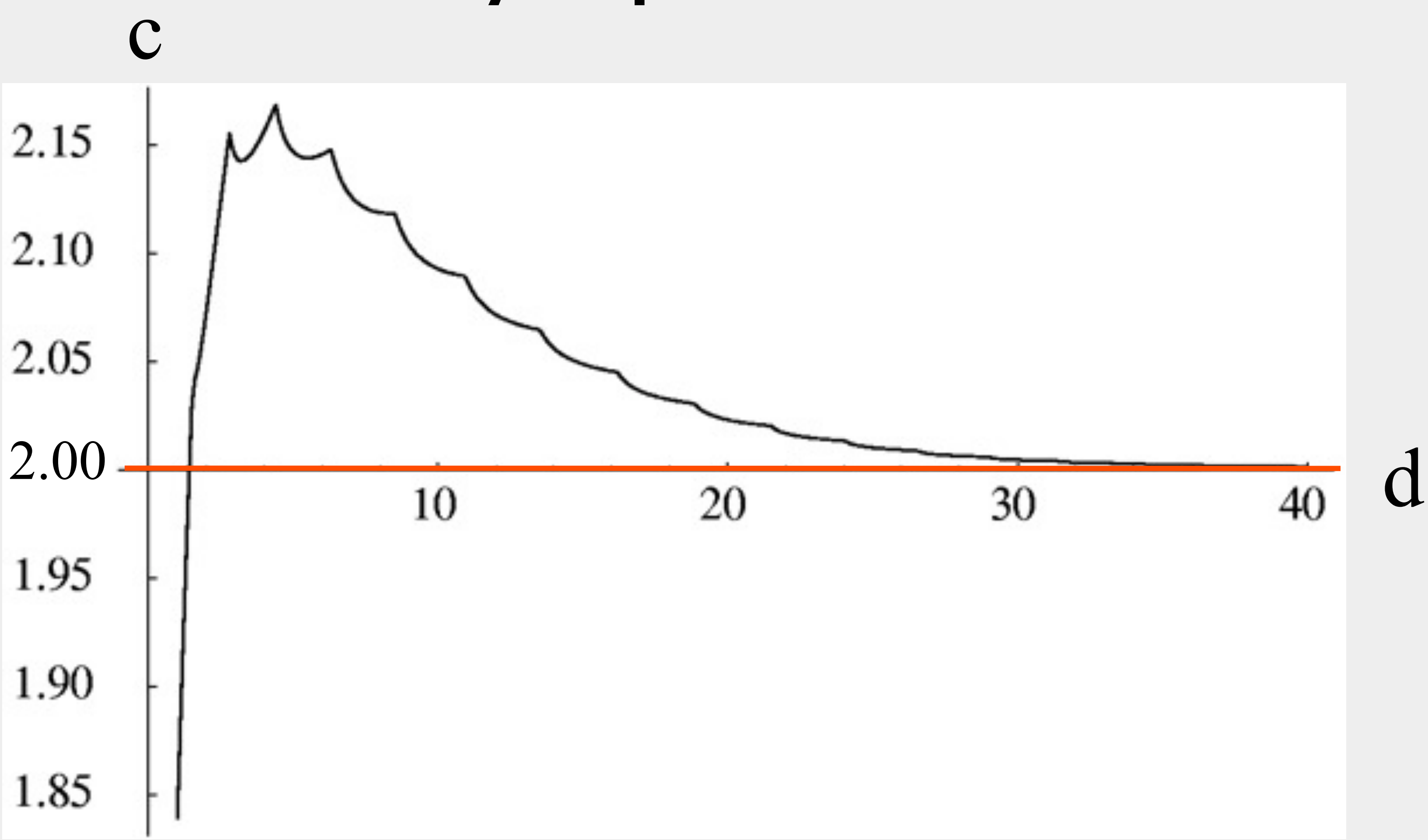
Asymptotics



Asymptotics



Asymptotics



An Upper Bound

Theorem: The circle strategy is asymptotically optimal.

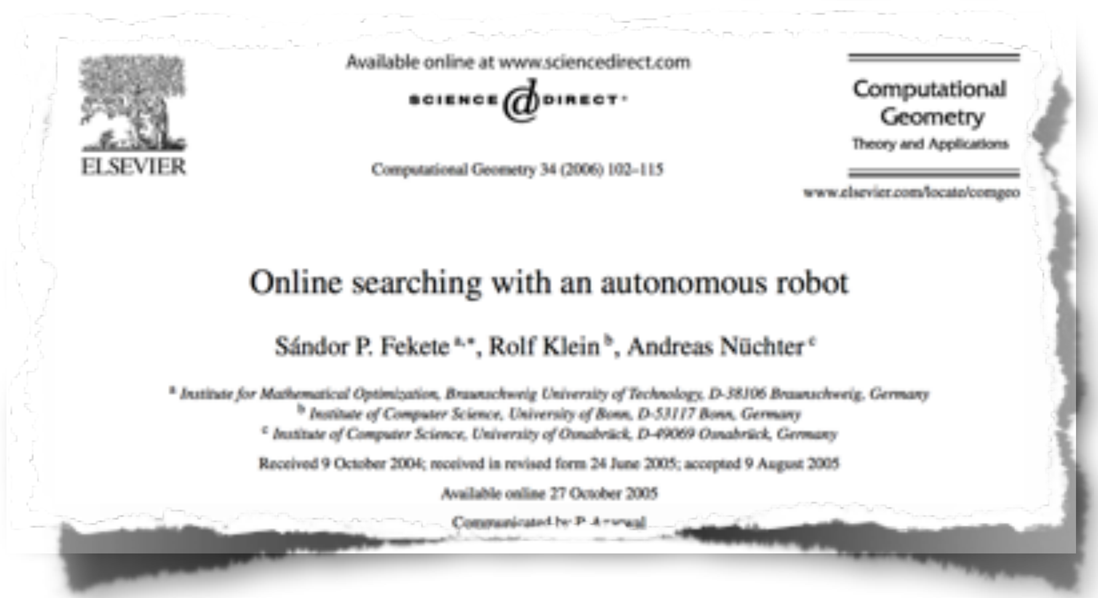
Sketch: Consider a factor $c = 2 + \delta$.

(1) Show that for large circle diameter, the step length grows expontially, as long as the direction is close to being orthogonal to the wall.

(2) Show that in this manner, a large step length can be reached. More specific, show that an average step length of at least 5 can be achieved at some point.

(3) Show that once the average step length is at least 5, it stays above 5. Thus, any necessary total distance can be traveled.

Practical Application


ELSEVIER
Available online at www.sciencedirect.com
SCIENCE DIRECT
Computational Geometry 34 (2006) 102–115
Computational Geometry
Theory and Applications
www.elsevier.com/locate/comgeo

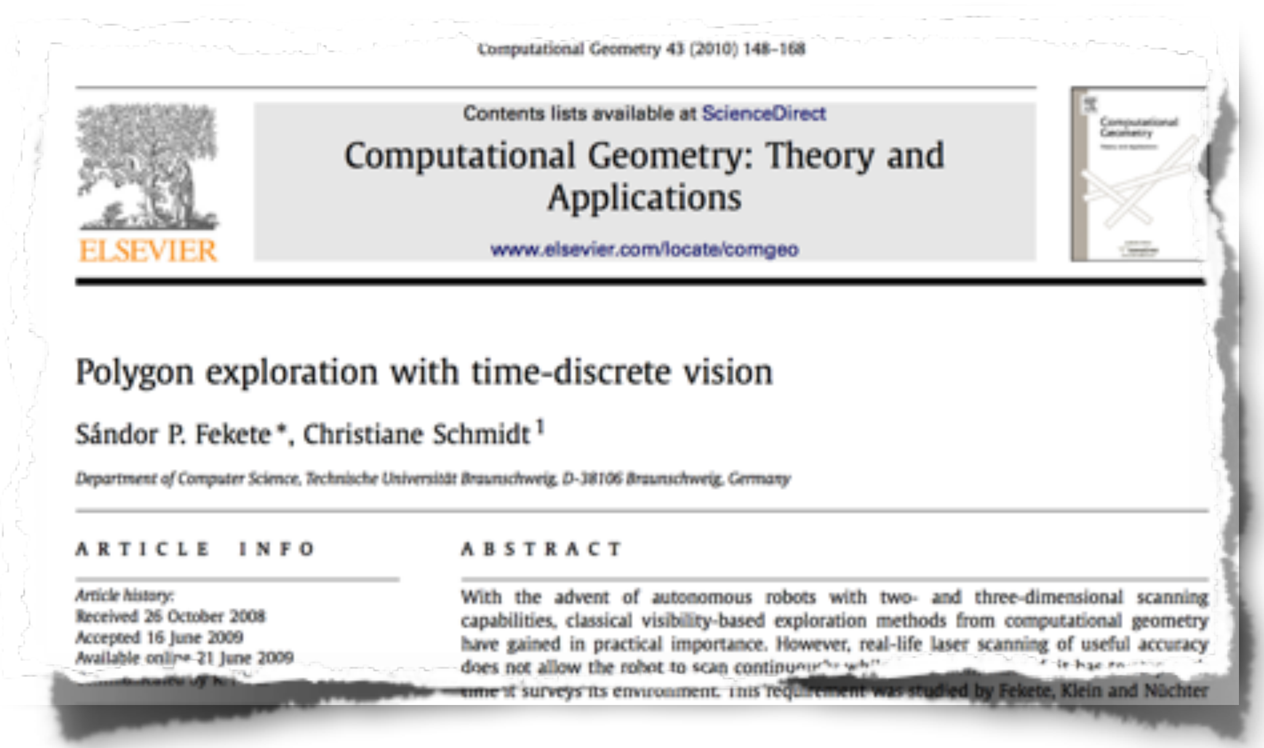
Online searching with an autonomous robot

Sándor P. Fekete [a,*], Rolf Klein [b], Andreas Nüchter [c]

[a] Institute for Mathematical Optimization, Braunschweig University of Technology, D-38106 Braunschweig, Germany
[b] Institute of Computer Science, University of Bonn, D-53117 Bonn, Germany
[c] Institute of Computer Science, University of Osnabrück, D-49069 Osnabrück, Germany

Received 9 October 2004; received in revised form 24 June 2005; accepted 9 August 2005
Available online 27 October 2005

Part 1.2:
Exploring rectilinear polygons



Computational Geometry 43 (2010) 148–168

Contents lists available at ScienceDirect

Computational Geometry: Theory and Applications

www.elsevier.com/locate/comgeo

ELSEVIER

Polygon exploration with time-discrete vision

Sándor P. Fekete *, Christiane Schmidt [1]

Department of Computer Science, Technische Universität Braunschweig, D-38106 Braunschweig, Germany

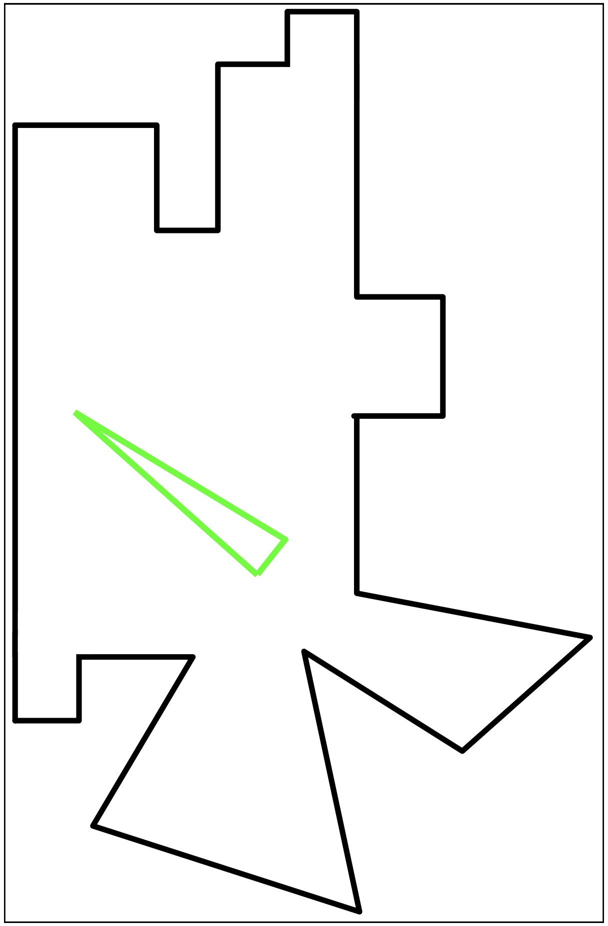
ARTICLE INFO

Article history:
Received 26 October 2008
Accepted 16 June 2009
Available online 21 June 2009

ABSTRACT

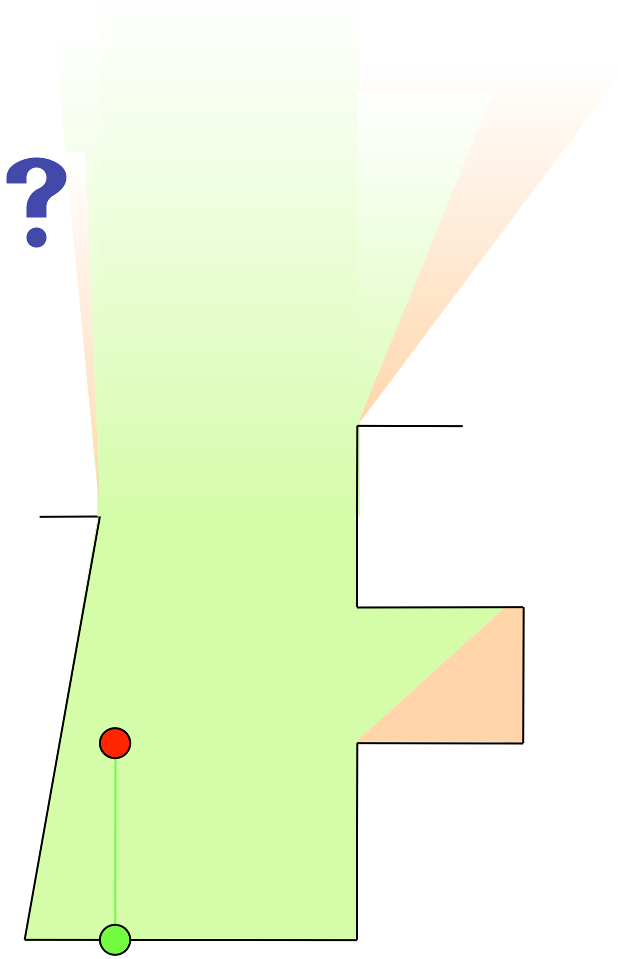
With the advent of autonomous robots with two- and three-dimensional scanning capabilities, classical visibility-based exploration methods from computational geometry have gained in practical importance. However, real-life laser scanning of useful accuracy does not allow the robot to scan continuously while ... time it surveys its environment. This requirement was studied by Fekete, Klein and Nüchter

Motivation



- Watchman problem
- Online, continuous vision:
  - optimum watchman route (L_1-metric) in simple rectilinear polygons (Deng et al.)
  - c=26.5 in simple polygons (Hoffmann et al.)
  - No competitive online algorithm for polygons with holes (Albers et al.)

Motivation



- Autonomous robot without continuous vision (scan costs)
- Watchman route
- Online problem
- Several classes of polygons
- Is it possible to achieve a competitive strategy?

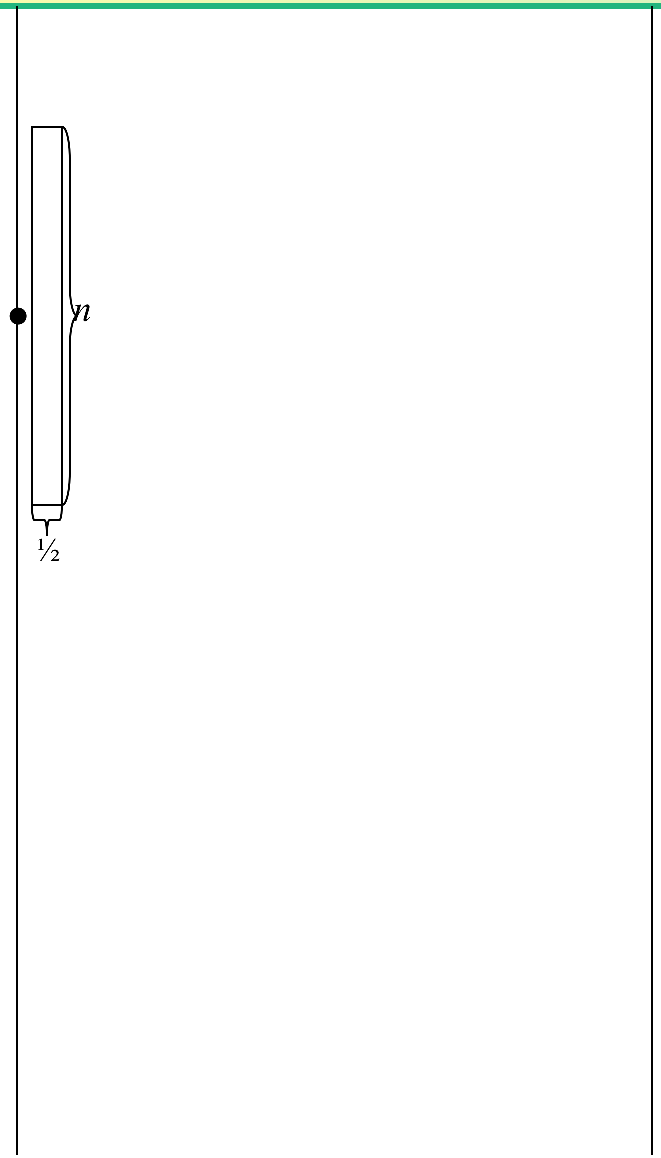
Polygons with Holes

Polygons with holes

Proposition:

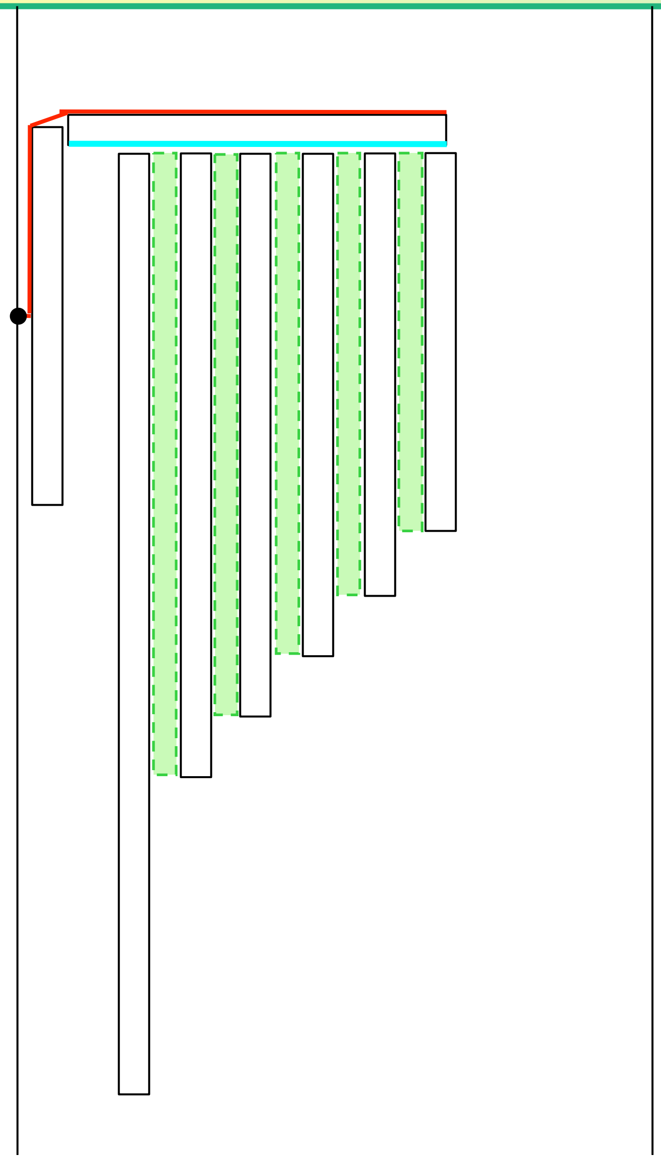
There is no strategy that achieves a bounded competitive ratio for the watchman problem with scan costs in case of a polygon with holes/obstacles.
This statement holds even if the polygon is rectilinear.

Proof of the proposition



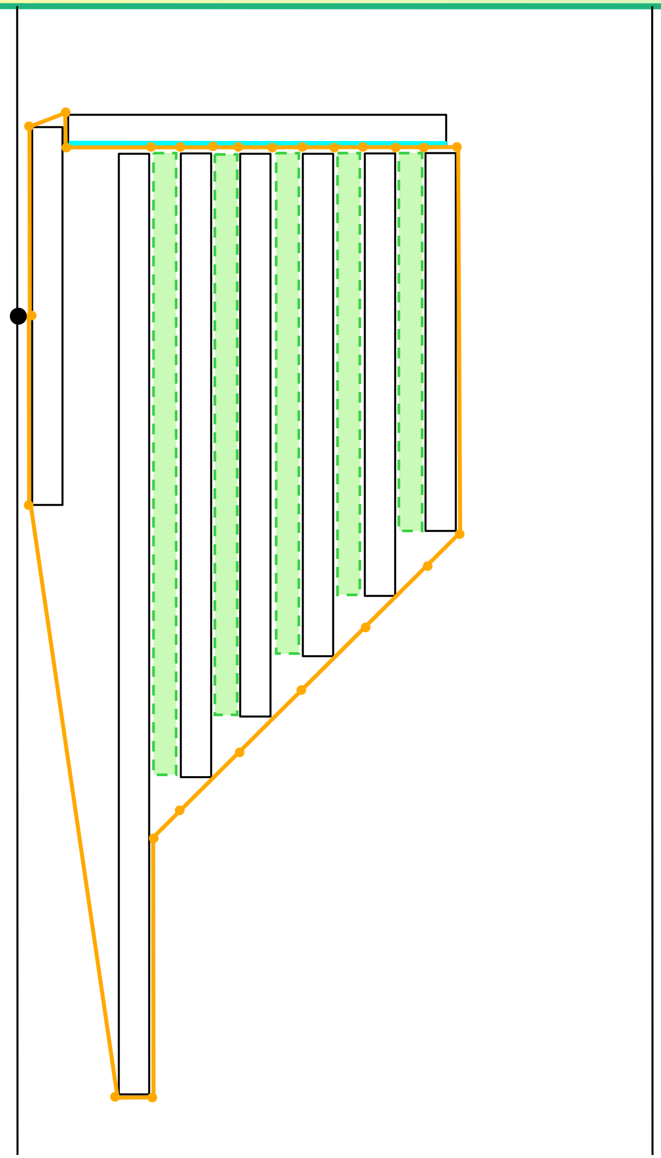
- Show: competitive ratio $\Omega(\sqrt{n})$
- Polygon with obstacles (panpipe)
- Further obstacles: placed depending on the strategy of the robot

Proof of the proposition



- Show: competitive ratio $\Omega(\sqrt{n})$
- Polygon with obstacles (panpipe)
- Further obstacles: placed depending on the strategy of the robot

Proof of the proposition



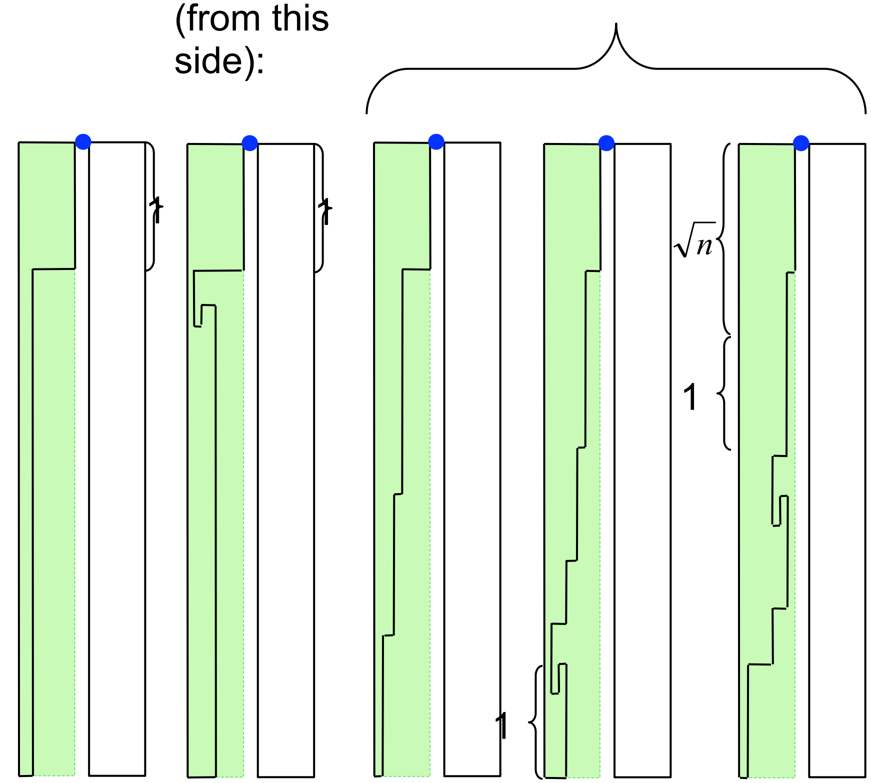
- Show: competitive ratio $\Omega(\sqrt{n})$
- Polygon with obstacles (panpipe)
- Further obstacles: placed depending on the strategy of the robot

Proof of the proposition

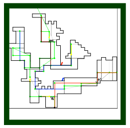
The robot traverses the row:

The robot does not turn into the row (from this side):

The robot walks into the row, but turns back:

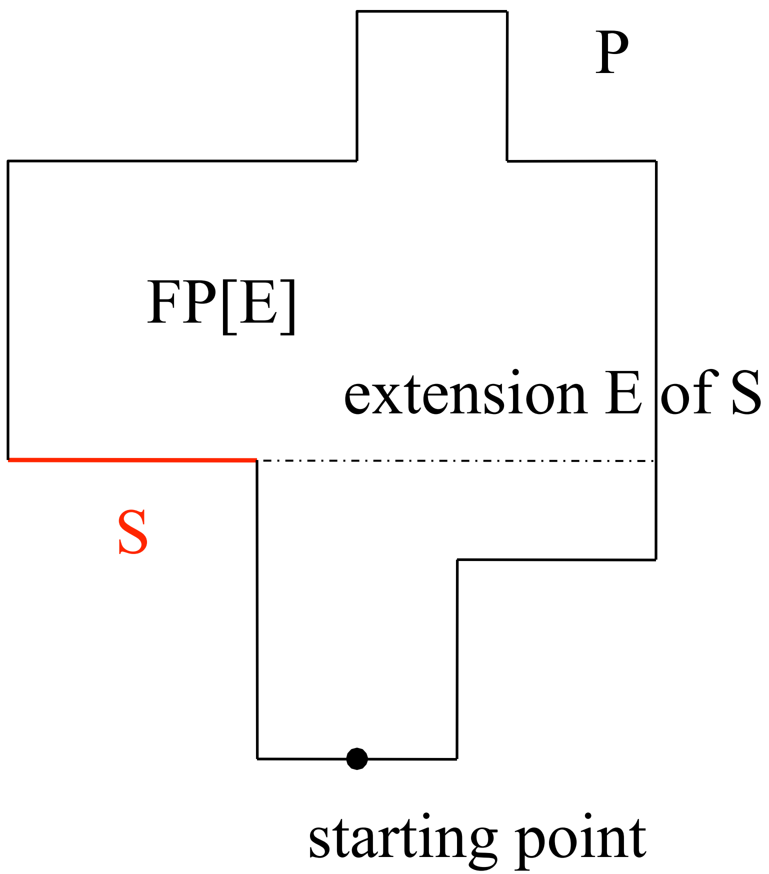


- The shape of the inserted objects depends on the path of the robot.



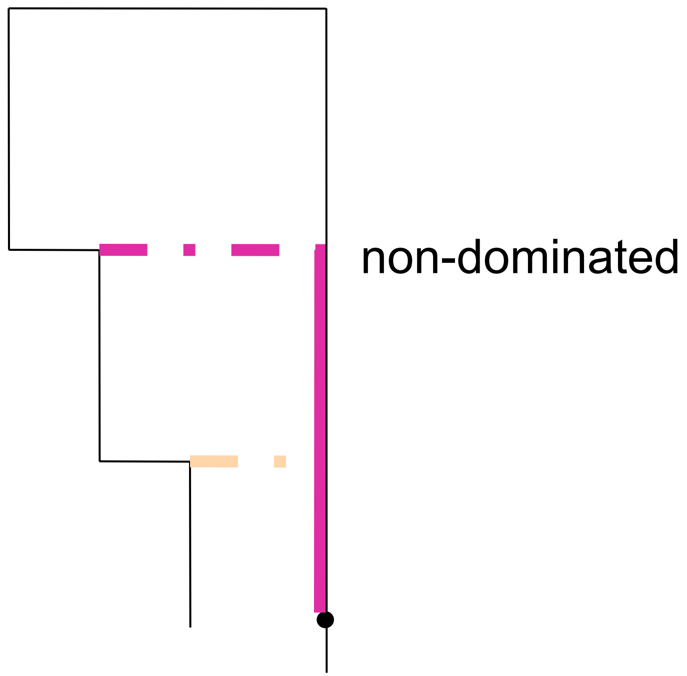
A Competitive Strategy for Simple Rectilinear Polygons

Extensions



- Two subpolygons
- Necessary and essential extensions

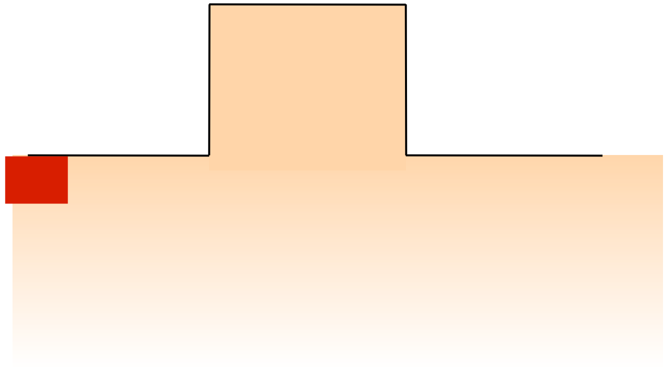
Extensions



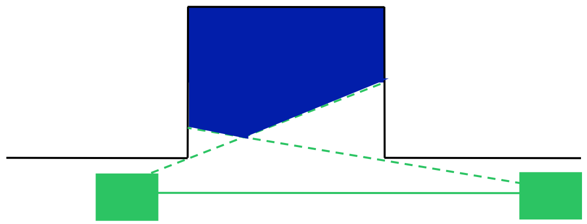
- Two subpolygons
- Necessary and essential extensions
- Advantage in rectilinear polygons

A competitive strategy for simple rectilinear polygons

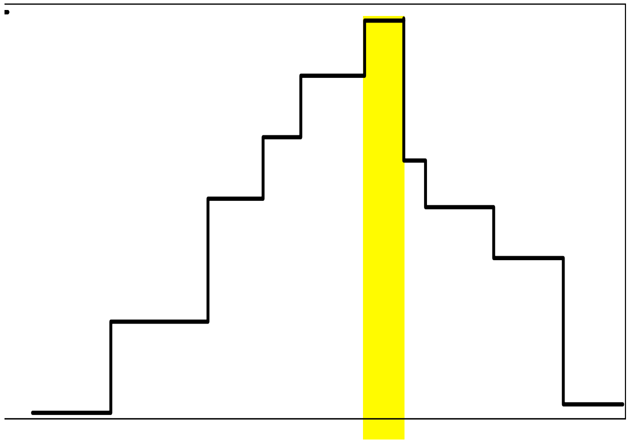
A competitive strategy for simple rectilinear polygons



A competitive strategy for simple rectilinear polygons

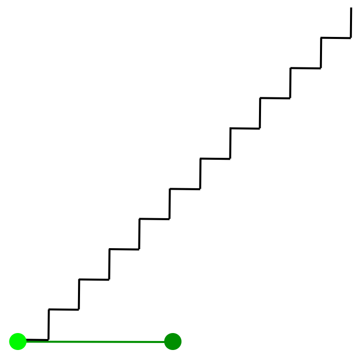


A competitive strategy for simple rectilinear polygons



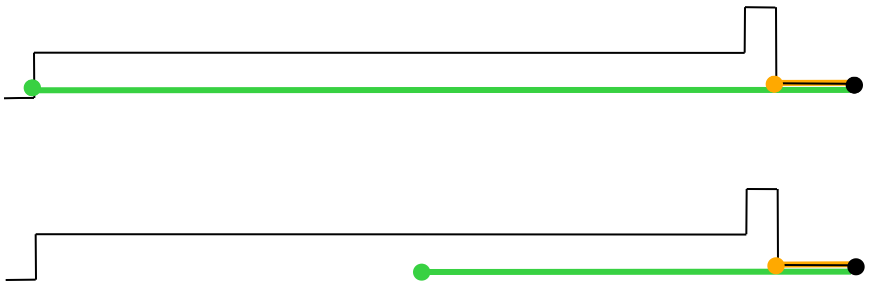
- Problem with niches
- It is necessary to limit the number of scan points

A competitive strategy for simple rectilinear polygons



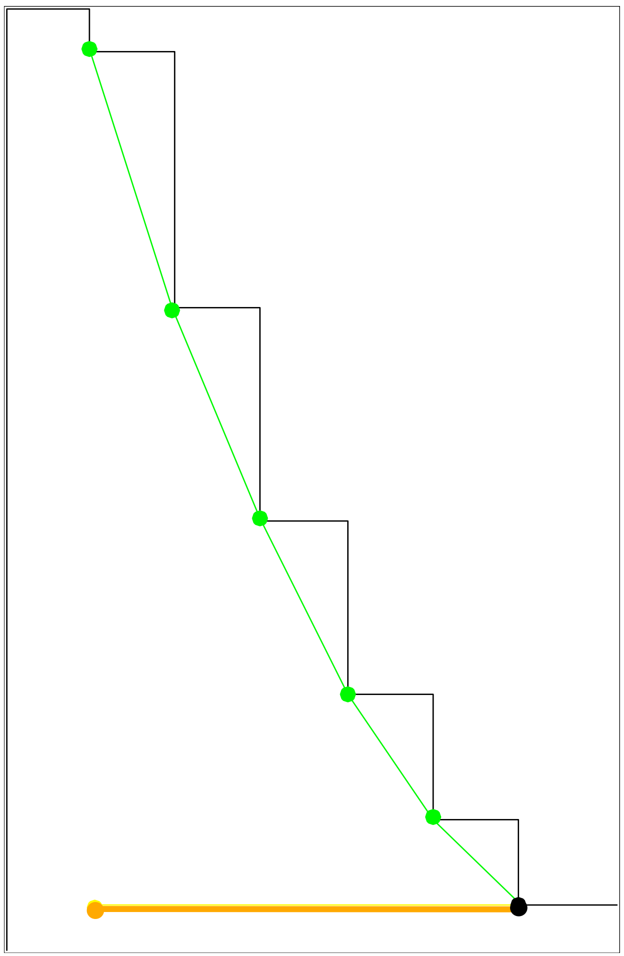
- Minimum side length a

A competitive strategy for simple rectilinear polygons



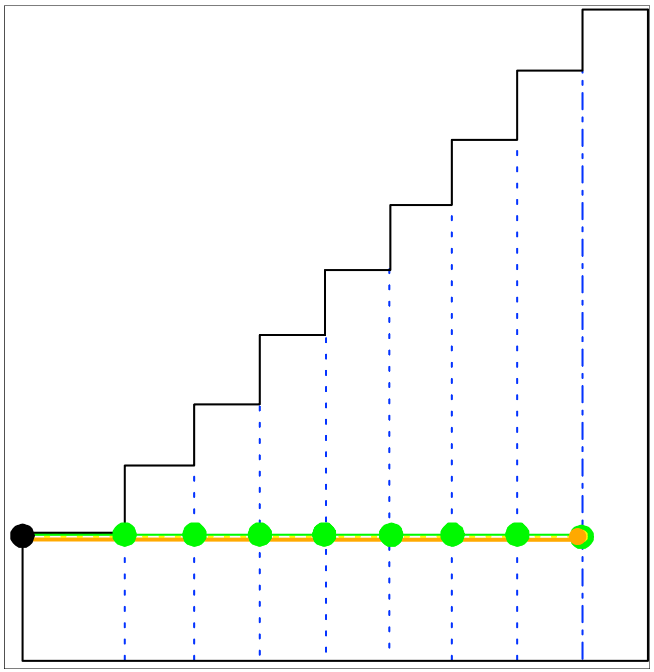
- Minimum side length a
- Consider the distance to the next corner (reflex vertex): walk beyond the corner if the distance to it is “short”

A competitive strategy for simple rectilinear polygons



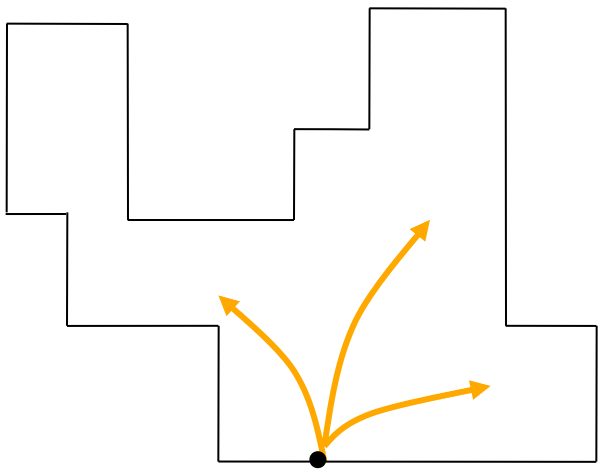
- Minimum side length a
- Consider the distance to the next corner (reflex vertex): walk beyond the corner if the distance to it is “short”
- Adapt the step length of the robot to the minimal necessary step length
- Move to the projection of a corner and not to the corner itself

A competitive strategy for simple rectilinear polygons



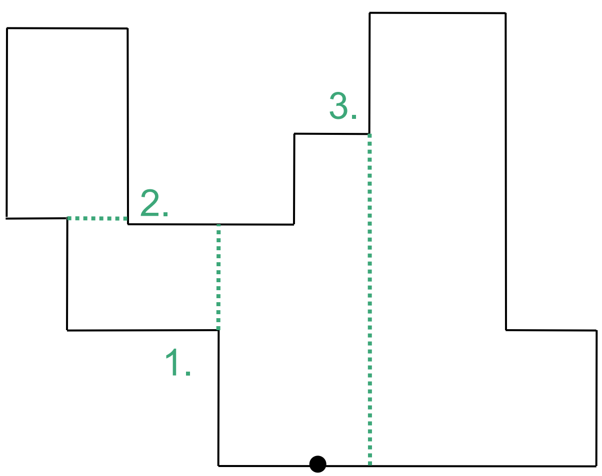
- Minimum side length a
- Consider the distance to the next corner (reflex vertex): walk beyond the corner if the distance to it is “short”
- Adapt the step length of the robot to the minimal necessary step length
- Move to the projection of a corner and not to the corner itself
- Do not scan on each necessary extension

Order of extensions



Optimum?

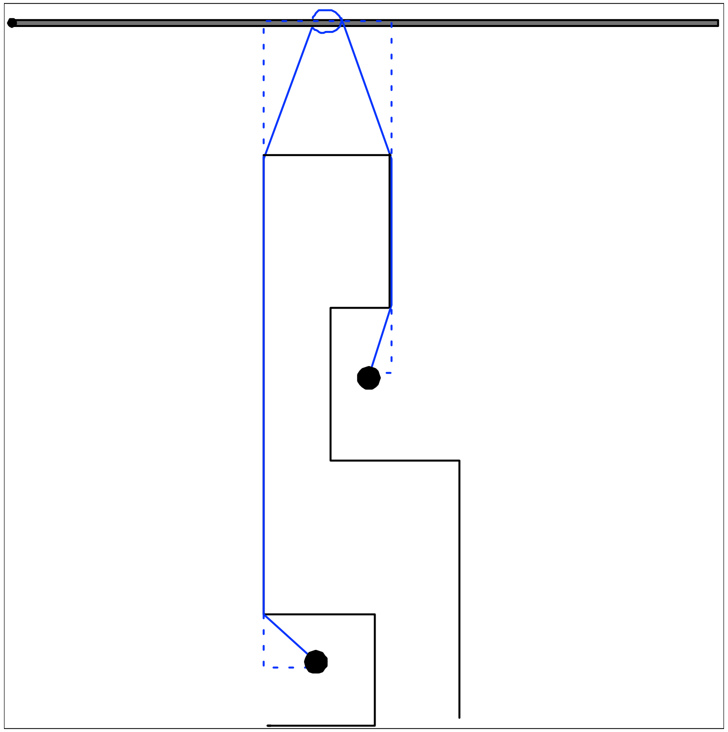
Order of extensions



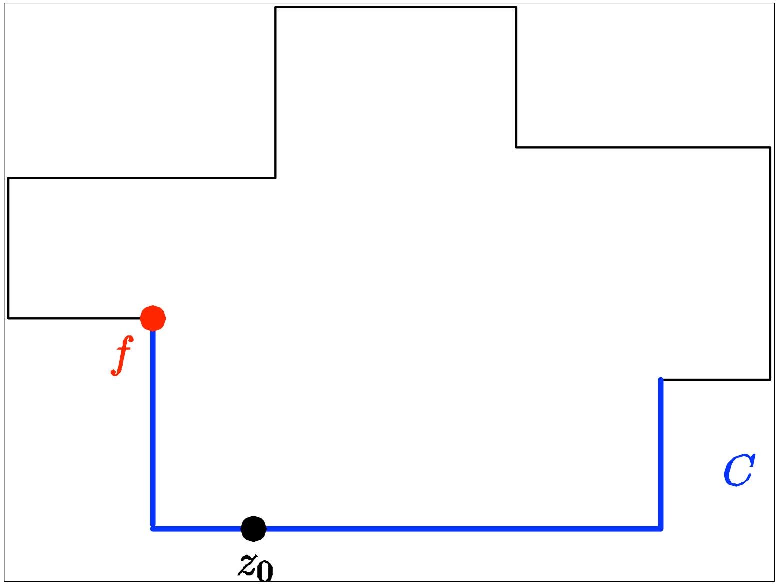
- GREEDY-ONLINE algorithm for a robot with continuous vision.
- Based on a proposition of Chin and Ntafos:

  Any optimum watchman route in P, a simple rectilinear polygon, will have to visit the essential edges in the order in which they appear on the boundary of P' (the new polygon obtained by removing the "non-essential" portions of the polygon).
- Transfer of this proposition.

GREEDY-ONLINE algorithm

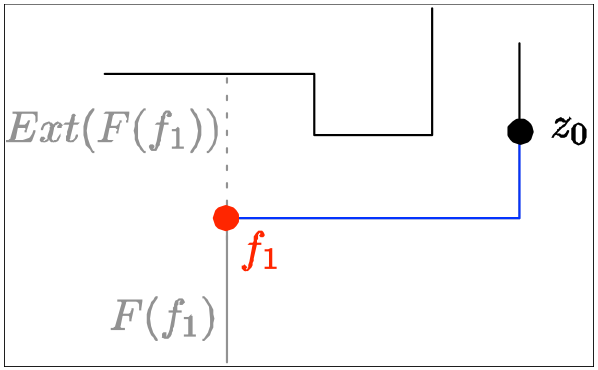


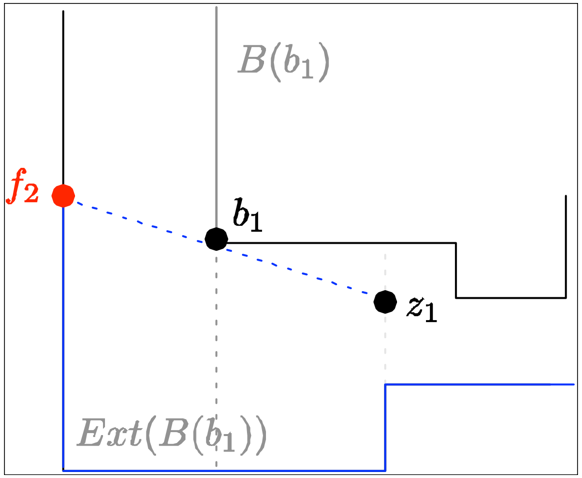
GREEDY-ONLINE algorithm



- Taut-Thread-Principle
- Consider the contiguous part of the boundary that was already visible from some point of the route
- Either f is a 270° corner or a corner blocks the sight such as only f^- is visible

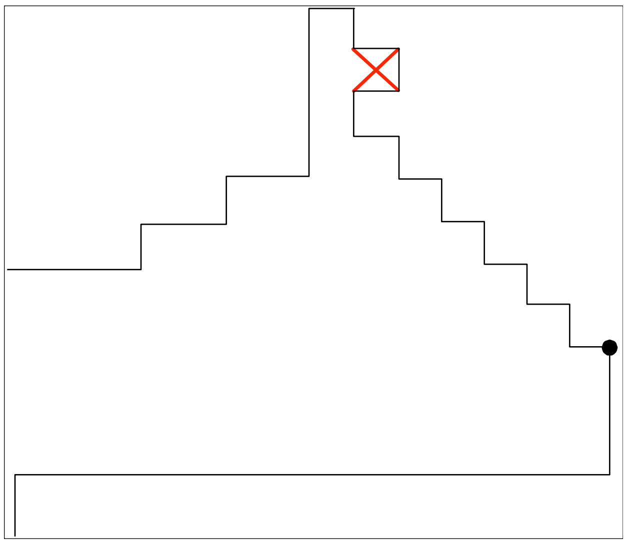
GREEDY-ONLINE algorithm





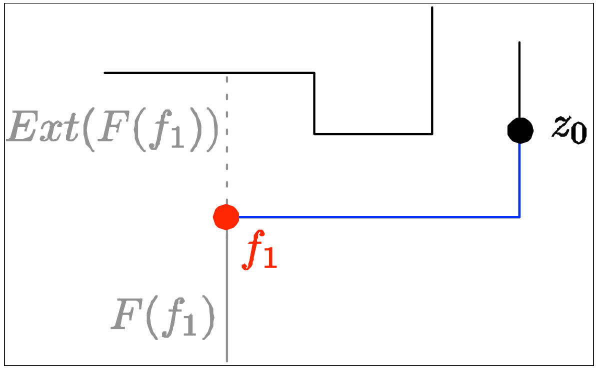
- Taut-Thread-Principle
- Consider the contiguous part of the boundary that was already visible from some point of the route
- Either f is a 270° corner or a corner blocks the sight such as only f^- is visible

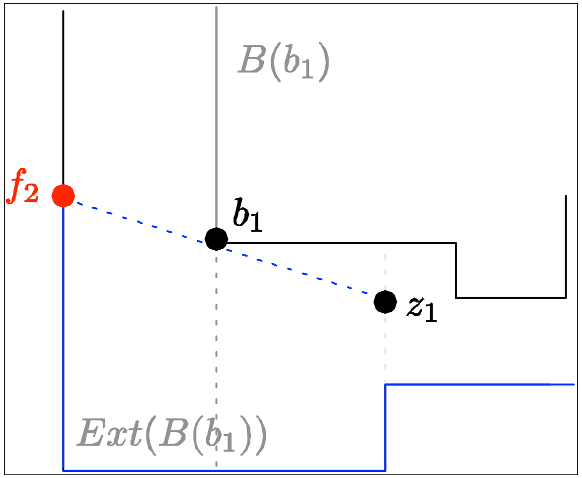
GREEDY-ONLINE algorithm



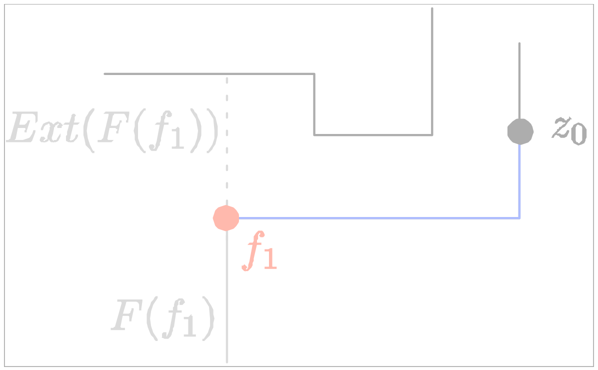
- Taut-Thread-Principle
- Consider the contiguous part of the boundary that was already visible from some point of the route
- Either f is a 270° corner or a corner blocks the sight such as only f^- is visible

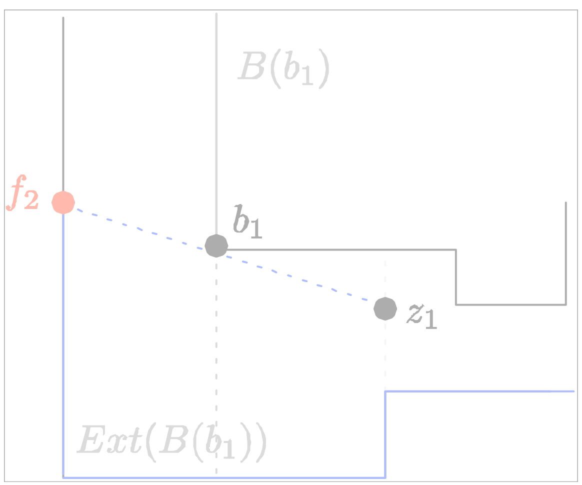
A competitive strategy for simple rectilinear polygons





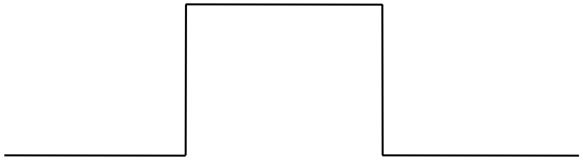
A competitive strategy for simple rectilinear polygons





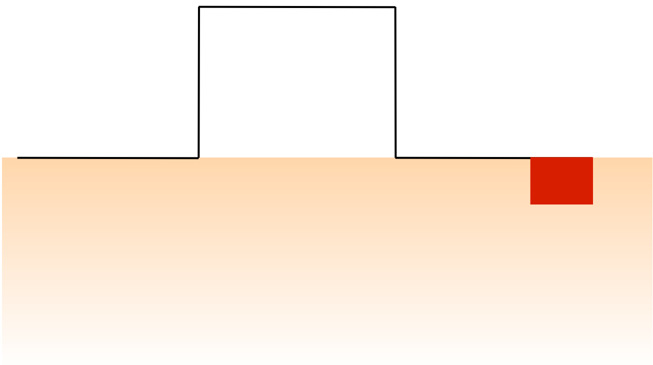
- Extensions of the GREEDY-ONLINE algorithm
- Interval case vs. extension case
- Reaching the extension on an axis-parallel path without a change of direction is possible/impossible
- In all cases of the case differentiation:
  - In case the robot runs beyond the extension: the robot is (is not) able to cover the total planned length
  - Positive line creation vs. negative line creation

Binary search in the strategy



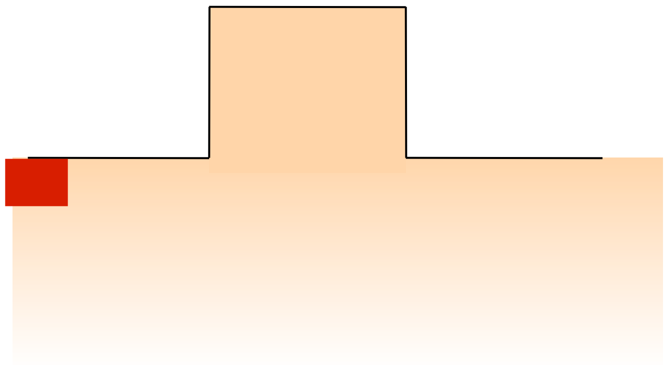
- Non-visible region (NVR): An area in which the parts of the boundary, which would be visible by simply passing them with continuous vision, are not yet completely visible.
- Discover passed non-visible regions with binary search.

Binary search in the strategy



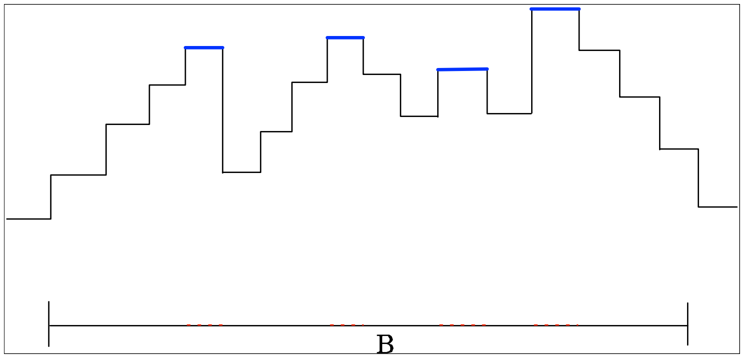
- Non-visible region (NVR): An area in which the parts of the boundary, which would be visible by simply passing them with continuous vision, are not yet completely visible.
- Discover passed non-visible regions with binary search.

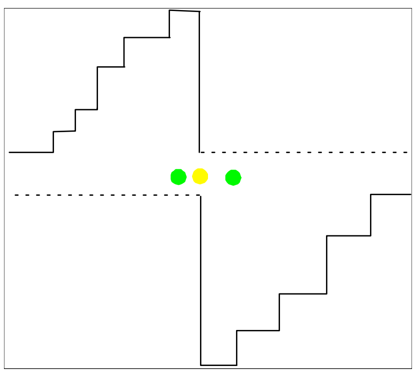
Binary search in the strategy



- Non-visible region (NVR): An area in which the parts of the boundary, which would be visible by simply passing them with continuous vision, are not yet completely visible.
- Discover passed non-visible regions with binary search.

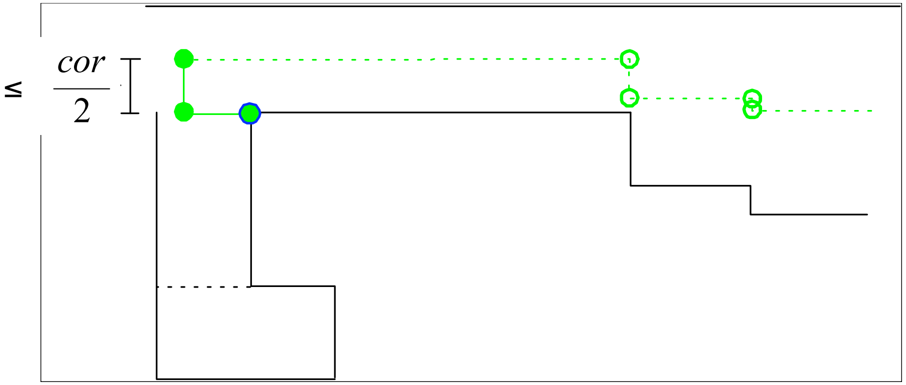
Binary search in the strategy





- If the optimum nees k scans in an interval, the robot which uses the strategy will need maximum
  - k binary searches (for each an upper bound is given) or
  - 2k binary searches if the NVRs may appear on two sides

Turn adjustments



- The optimum may have the opportunity to turn off before the robot, following the strategy, does.
- The robot may discover a corridor inside a non-visible region.
- ➢ Adjustments to have the best basic position for the next turn
- Minimum corridor width a_k

The strategy

$a \leq 1$:

A. An axis-parallel move to E is possible without a turn

- $e \geq 2a+1$: interval case
  Let d_i be the actual distance to the perpendicular of the next counterclockwise extension
  – If $d_i > 2a+1$, move to the perpendicular of the corner
  – If $d_i \leq 2a+1$: If $d_i > a$: cover a distance of $2d_i+1$
    If $d_i \leq a$: cover a distance of $2a+1$

    Apply binary search if necessary, that means, if non-visible regions appear.
  – If no corner appears on the counterclockwise side, move directly to E.

  In case we run beyond E with a step of length $2d_i+1/2a+1$:
  i. If we do not cover the total distance, because of the boundary: Run as far as possible, go back to E, move back in steps of length 1, apply binary search for NVRs (on the counterclockwise side till E, on both sides beyond E) and if a corridor is identified, use it and make turn adjustments
  ii. If we may cover the total distance:
     I. negative line creation: Apply binary search, if a corridor is discovered inside a NVR, use it and make turn adjustments.
     II. Positive line creation: Go back to E, move back in steps of length 1, apply binary search and search for a corridor and the critical extension, make turn adjustments.
- $e < 2a+1$: extension case
  Cover a distance of 2e+1. In case:..(i., ii.)

The strategy

$a \leq 1$:

A. An axis-parallel move to E is possible without a turn

- e≥2a+1: interval case
- e<2a+1: extension case

B. An axis-parallel move to E is not possible without a change of direction: Let b_i be the distance to the sight-blocking corner.

- e ≥a+1: interval case
  - No non-visible region up to the sight-blocking corner
  - Along the boundary up to the sight-blocking corner occur non-visible regions
- e<a+1: extension case

$a > 1$:

Similar; with scans every time a distance of a is covered.

The strategy

$a \leq 1$:

A. An axis-parallel move to E is possible without a turn

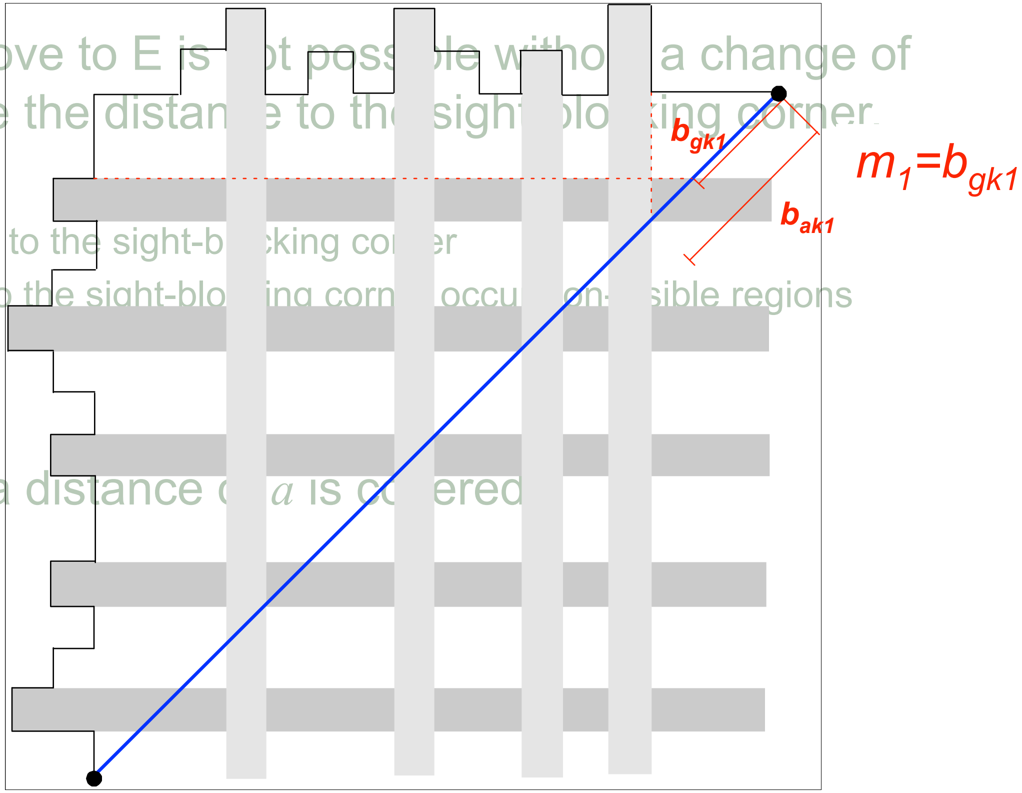
- e≥2a+1: interval case
- e<2a+1: extension case

B. An axis-parallel move to E is ot poss le witho a change of direction: Let b_i be the distan e to th sight blo king corner

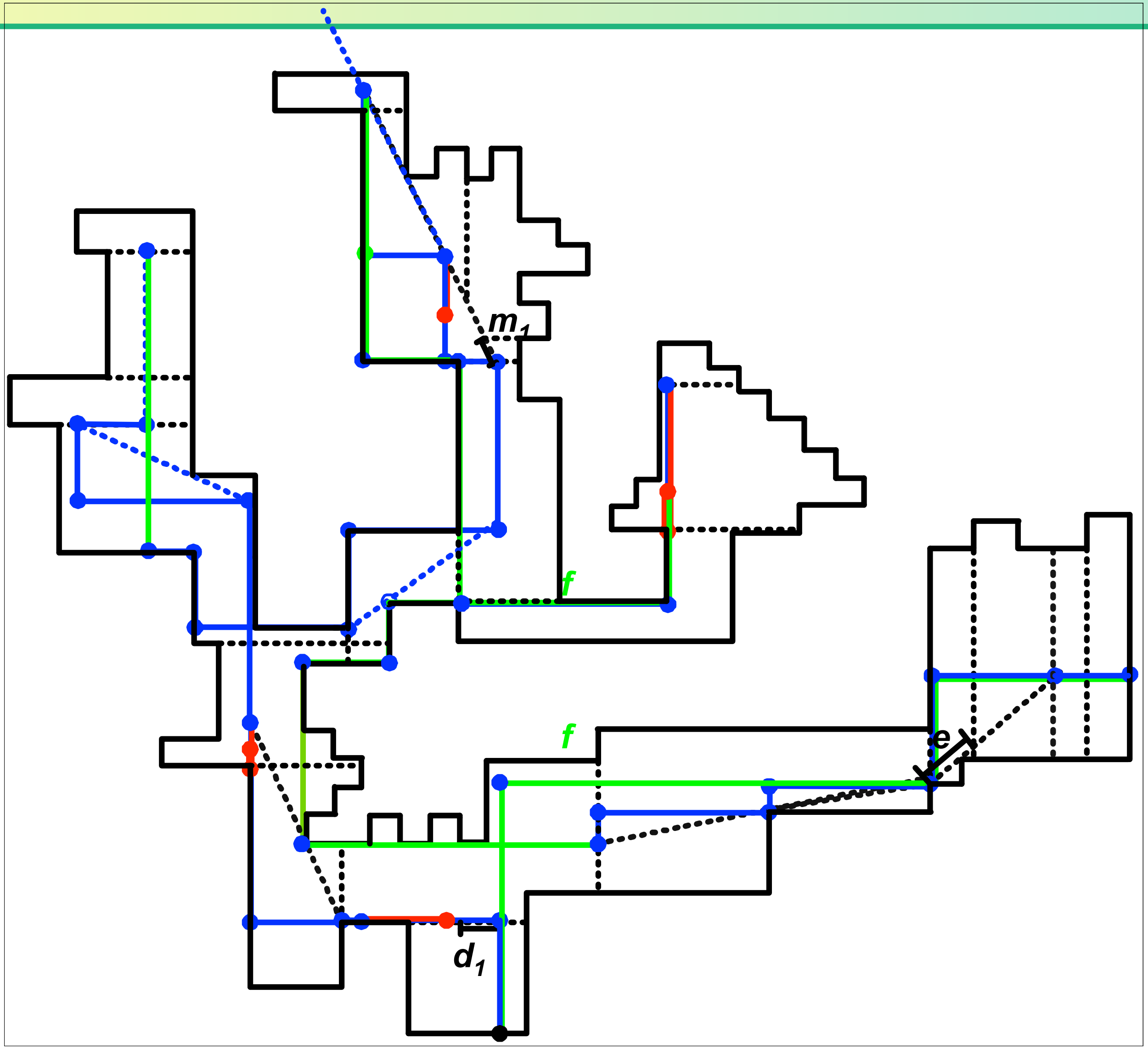
- e ≥a+1: interval case
  - No non-visible region up to the sight-b cking co er
  - Along the boundary up to the sight-blo ing corn occu on- sible regions
- e<a+1: extension case

a > 1:

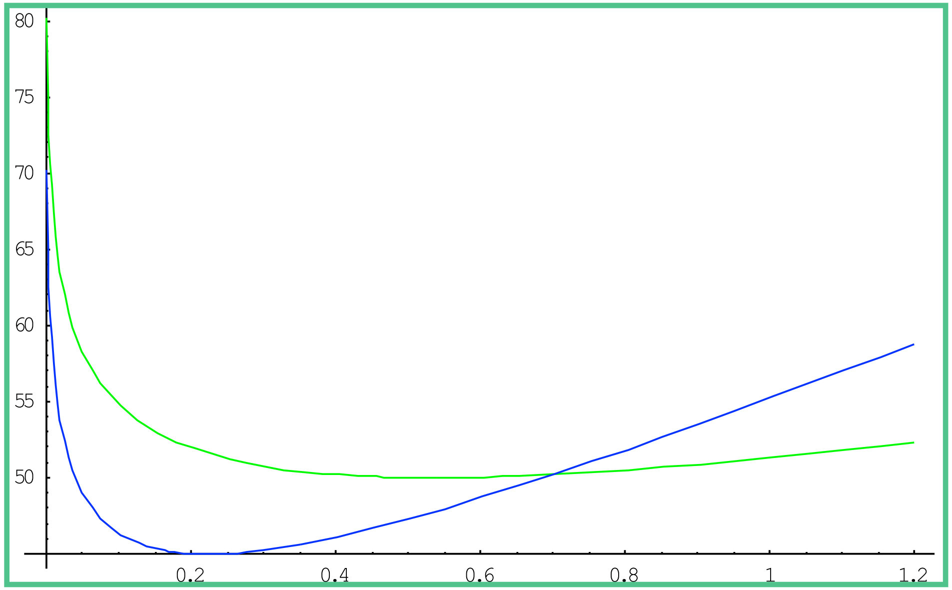
Similar; with scans every time a distance a is co ered



An example



The competitive ratio of the strategy



a	upper bound for c
1	55.2294
0.8	51.8168
0.7	50.2083
0.5	50.0000
0.1	54.8000
0.01	67.0336
0.0001	93.4919
0.000001	120.0661

- If we assume $a = a_k$:

$$c \le \begin{cases} 8a + 34 + 4\dfrac{\ln\left(\dfrac{2a+3}{a}\right)}{\ln(2)}, & 0 \le a < 0.70043 \\ 20a + 24 + 4\dfrac{\ln\left(\dfrac{4a+3}{a}\right)}{\ln(2)}, & 0.70043 \le a \le 1 \end{cases}$$

Computational Geometry 43 (2010) 148–168

Contents lists available at ScienceDirect

Computational Geometry: Theory and Applications

www.elsevier.com/locate/comgeo

ELSEVIER

Polygon exploration with time-discrete vision

Sándor P. Fekete *, Christiane Schmidt [1]

Department of Computer Science, Technische Universität Braunschweig, D-38106 Braunschweig, Germany

ARTICLE INFO

Article history:
Received 26 October 2008
Accepted 16 June 2009
Available online 21 June 2009

ABSTRACT

With the advent of autonomous robots with two- and three-dimensional scanning capabilities, classical visibility-based exploration methods from computational geometry have gained in practical importance. However, real-life laser scanning of useful accuracy does not allow the robot to scan continuously ... time it surveys its environment. This requirement was studied by Fekete, Klein and Nüchter

Part 1.3:
Searching with turn cost

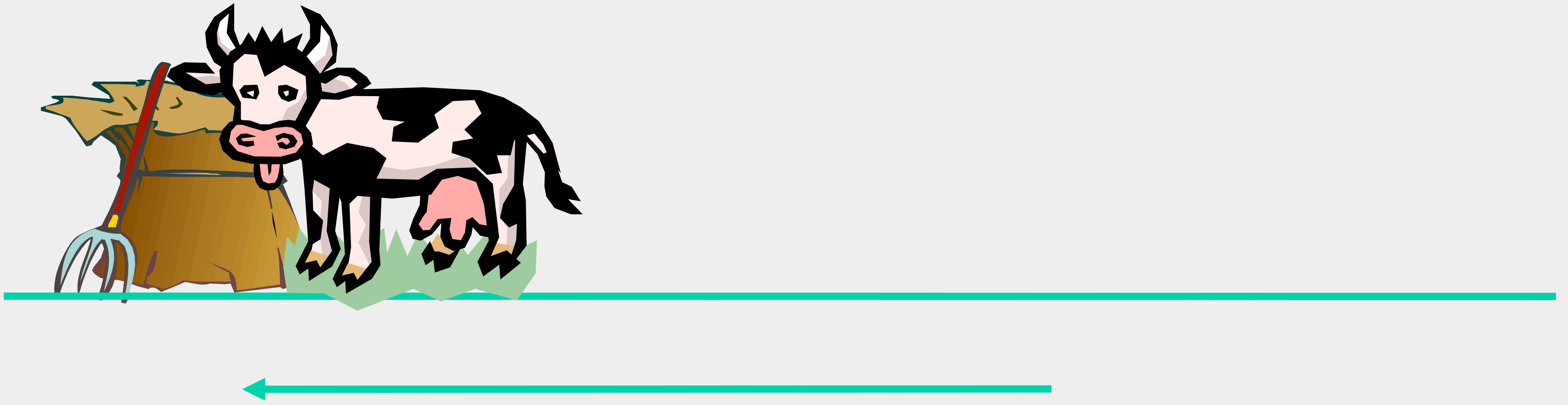
Available online at www.sciencedirect.com
ScienceDirect
Theoretical Computer Science 361 (2006) 342–355
Theoretical Computer Science
www.elsevier.com/locate/tcs

Online searching with turn cost

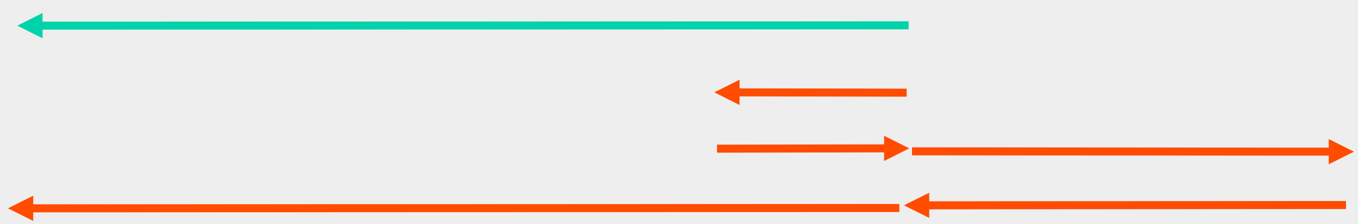
Erik D. Demaine[a], Sándor P. Fekete[b,*], Shmuel Gal[c]

[a]Computer Science and Artificial Intelligence Laboratory, MIT, Cambridge MA, USA
[b]Department of Mathematical Optimization, Braunschweig University of Technology, Braunschweig, Germany
[c]Department of Statistics, University of Haifa, Haifa, Israel

Online Searching



Online Searching



Template: "Math Prog Talk"

Technische Universität Braunschweig

Linear Search

GIVEN: A starting position 0 on a line.

MISSION: Find an object at an unknown location.

UNKNOWN: (1) Direction of the object
(2) Distance OPT of the object

WANTED! A competitive strategy for the searcher that will guarantee that the object is found in time at most c · OPT for some constant "competitive" factor c.

Literature

BELLMAN 1963: Introduced the problem

BECK and NEWMAN 1970: Solved the problem

GAL 1974: Solved a generalization:

Search on m rays
Optimal competitive ratio: $1 + \frac{2m^m}{(m-1)^{m-1}}$
Optimal strategy: Geometric series with ratio $\left(\frac{m}{m-1}\right)$

Technische Universität Braunschweig

Literature

KAO Also known as the cow-path problem

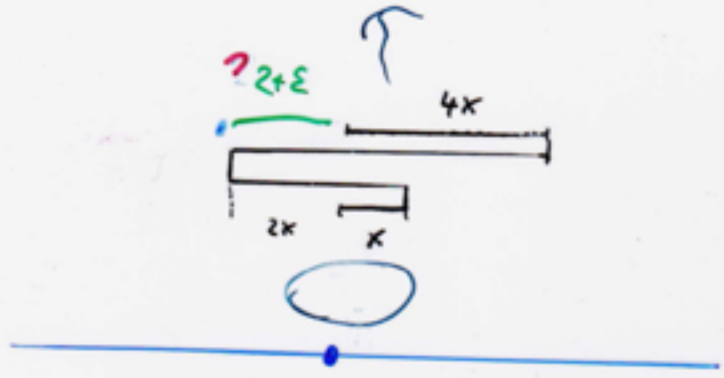
GAL 1980: Optimal trajectory to this type of problem is always a geometric series

BAEZA-YATES, CULBERSON, RAWLINS 1988: (and various others independently) Rediscovered problem and solution

Many variations and applications, in particular for geometric searching.

Doubling

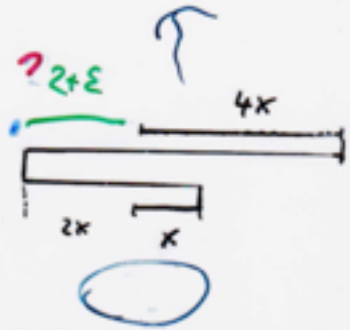
Keep doubling the search distance before returning:



Known: This guarantees a competitive factor of 9, and this is best possible!

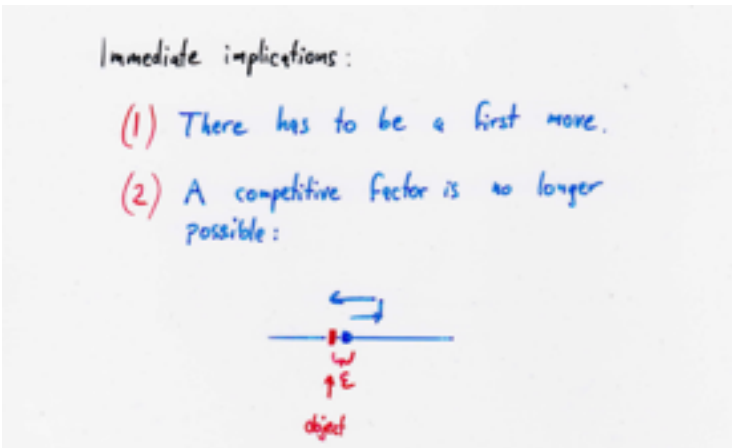
Doubling

Keep doubling the search distance before returning:



DISADVANTAGE: There is no real "start" of the trajectory - it's just a geometric series, and each previous step was half as long as the latest one!

Turn Cost

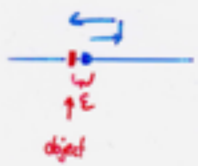

Immediate implications:
(1) There has to be a first move.
(2) A competitive factor is no longer possible:
ε
object

Turn Cost

Immediate implications:

(1) There has to be a first move.

(2) A competitive factor is no longer possible:

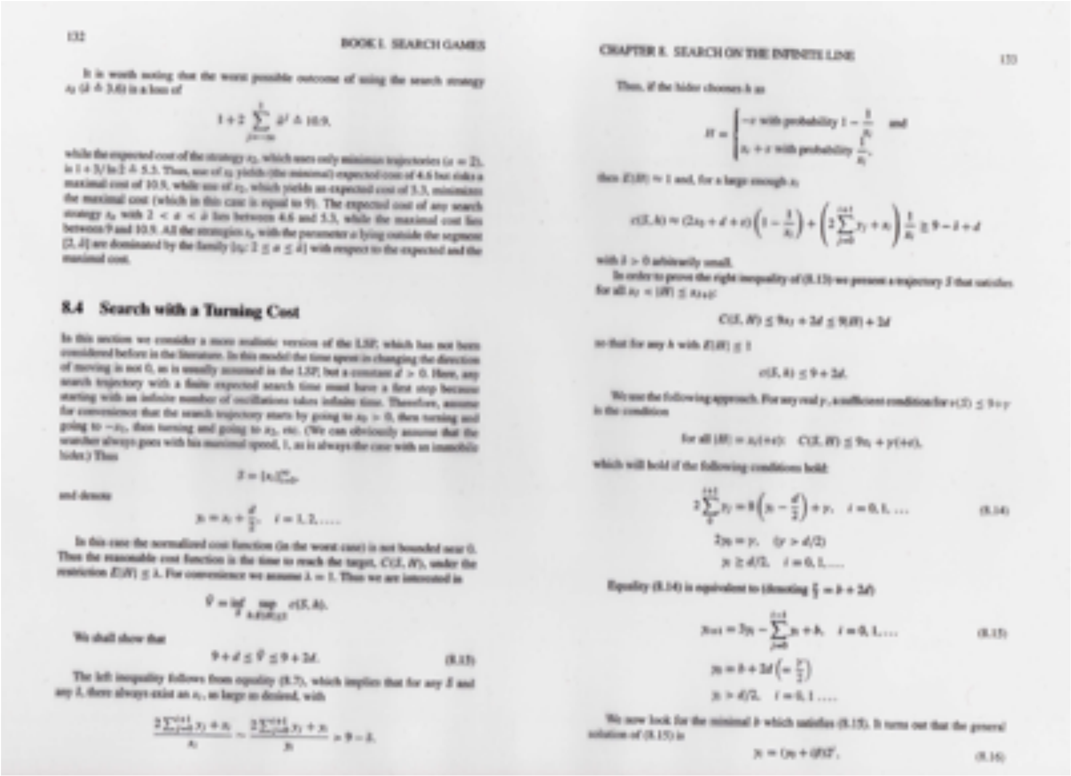


Searching in the wrong direction takes at least one turn, for a cost of d, compared to optimal ε

Fix: Consider $c \cdot OPT + f(d)$

- and possibly $c \cdot OPT + \lambda \cdot d$

An Open Problem


132 BOOK I. SEARCH GAMES

8.4 Search with a Turning Cost

CHAPTER 8. SEARCH ON THE INFINITE LINE 133

An Open Problem

134 BOOK I. SEARCH GAMES

where $\beta \geq 0$ is a nonnegative parameter. (Because by (8.15) $y_{i+1} - y_i = 3y_i - 4y_{i-1}$, denoting $y_i = 2^i \alpha_i$ it easily follows that $\alpha_{i+1} - \alpha_i = \alpha_i - \alpha_{i-1}$, which leads to (8.16).)

Using (8.16) for $i = 0, 1$ in (8.15) it follows that $\beta = y_0 - d$. Since $\beta \geq 0$ and $\gamma = 2y_0$, it easily follows that $\gamma \geq 2d$. On the other hand, the value $9 + 2d$ can be achieved by the following trajectory

$$y_i = d2^i, \qquad x_i = d2^i - d/2, \quad i = 0, 1, \dots$$

with the time to reach $x_i + \varepsilon$ being (neglecting $O(\varepsilon)$)

$$2\sum_0^{i+1} y_i + x_i = 2d(2^{i+2} - 1) + d2^i - d/2 = 9x_i + 2d.$$

Since $E|H| \leq 1$, the last equation guarantees expected time not exceeding $9 + 2d$.

Is $9 + 2d$ the best possible constant? This is still an open problem. (Note that (8.14) is a sufficient but not a necessary condition.)

Positions

The factor c can be at best 9!

(→ Consider d arbitrarily small compared to OPT.)

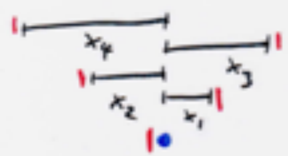
Suppose the searcher moves

x_1 to the right and returns,

x_2 to the left and returns,

x_3 to the right

(etc.)



Positions

The factor c can be at best 9!

(→ Consider d arbitrarily small compared to OPT.)

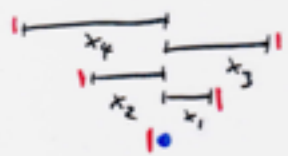
Suppose the searcher moves

x_1 to the right and returns,

x_2 to the left and returns,

x_3 to the right

(etc.)



Critical positions for hiding:

$y_0 = -\varepsilon$

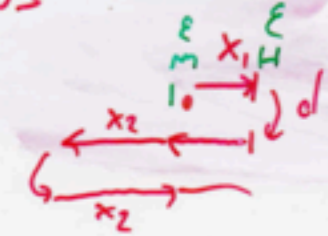
$y_1 = x_1 + \varepsilon$

$y_2 = -x_2 - \varepsilon$

$y_3 = x_3 + \varepsilon$

(etc.)

MORE CONDITIONS



y_0 must be reached in time:

$$2x_1 + d + \varepsilon \leq 9\varepsilon + \lambda d$$

y_1 must be reached in time:

$$2x_1 + 2x_2 + 2d + x_1 + \varepsilon \leq 9(x_1 + \varepsilon) + \lambda d$$

y_2 :

$$2x_1 + 2x_2 + 2x_3 + 3d + x_2 + \varepsilon \leq 9(x_2 + \varepsilon) + \lambda d$$

y_n :

$$2x_1 + \ldots + 2x_{n+1} + (n+1)d \leq 8x_n + \lambda d$$

This must hold for all $\varepsilon > 0$, so we get

An Infinite LP

$$\min \lambda$$

$$\begin{aligned}
2x_1 \qquad\qquad\qquad\qquad\qquad\qquad & + d \le \lambda d \\
2x_1 + 2x_2 \qquad\qquad\qquad\qquad\quad & + 2d \le 8x_1 + \lambda d \\
2x_1 + 2x_2 + 2x_3 \qquad\qquad\quad & + 3d \le 8x_2 + \lambda d \\
\vdots \qquad \vdots \qquad \vdots \qquad \ddots \qquad & \quad \vdots \qquad \vdots \\
2x_1 + 2x_2 + 2x_3 \quad \cdots \quad + 2x_{n+1} & + (n+1)d \le 8x_n + \lambda d \\
\vdots \qquad \vdots \qquad \vdots \qquad \ddots \qquad & \quad \vdots \qquad \vdots \\
& x_i \ge 0
\end{aligned}$$

(1) Infinite primal optimal solution describes optimal strategy of searcher.

(2) Optimal λ is tight value of turn cost penalty.

(3) Infinite dual optimal solution gives explicit proof of tightness.

Solving the Infinite LP

SOLVING SUBSYSTEMS

Only using the first n constraints yields a relaxation, with solutions $x_i^{(n)}$ and λ_n. Each λ_n is a lower bound for λ.

Approach:

(1) Run CPLEX on subsystems

(2) Consider convergence of solutions

(3) Construct infinite solution

(4) Verify solution

2	1.2500	0.1250	0.0000			0.7500	0.2500		
3	1.4166	0.2083	0.3333	0.0000		0.6666	0.2500	0.0833	
4	1.5313	0.2656	0.5625	0.6875	0.000	0.6250	0.2500	0.0937	0.0312
5	1.6125	0.3062	0.7250	1.1750	1.3000	0.6000	0.2500	0.1000	0.0375
10	1.8001	0.4000	1.1003	2.3001	4.3031	0.5500	0.2500	0.1125	0.0500
20	1.9000	0.4500	1.3000	2.9000	5.700	0.5250	0.2500	0.1187	0.0562
40	1.9500	0.4750	1.4000	3.2000	6.4333	0.5125	0.2500	0.1218	0.0593
50	1.9600	0.4800	1.4200	3.2600	6.8600	0.5100	0.2500	0.1225	0.0600
100	1.9800	0.4900	1.4600	3.3800	7.1600	0.5050	0.2500	0.1237	0.0612
200	1.9900	0.4950	1.4800	3.4400	7.3400	0.5025	0.2500	0.1243	0.0618
400	1.9950	0.4975	1.4900	3.4700	7.4200	0.5012	0.2500	0.1245	0.0621

Observe : Logarithmic convergence

Guess :

∞ 2.0000 0.5000 1.5000 3.5000 7.5000 0.5000 0.2500 0.1250 0.0625

Verifying the Solution

Choose: $x_i = \left(2^i - \frac{1}{2}\right) d$

$c_j = \frac{1}{2^j}$

Check primal solution, i.e. search strategy:

Verifying the Solution

Choose: $x_i = \left(2^i - \frac{1}{2}\right) d$

$c_i = \frac{1}{2^i}$

Check primal solution, i.e. search strategy:

Inequality n yields

$$\sum_{i=1}^{n+1} 2(x_i) - 8x_n + (n+1)d \leq \lambda d$$

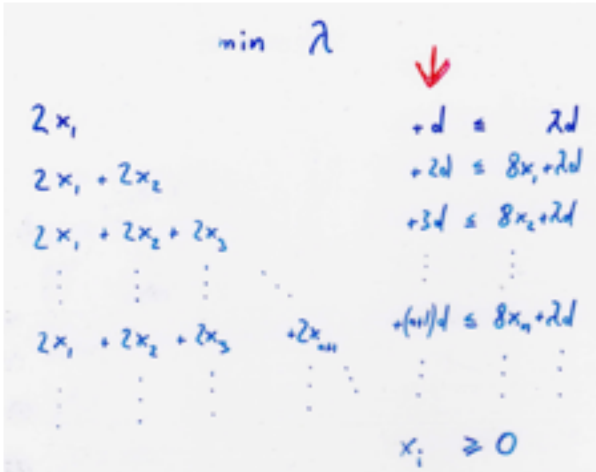
or $$\sum_{i=1}^{n+1} 2\left(2^i - \cancel{\tfrac{1}{2}}\right)d - 8\left(2^{n-1} - \tfrac{1}{2}\right)d + \cancel{(n+1)d} \leq \lambda d$$

or $$2^{n+2} - 2 \quad - 2^{n+2} + 4 \leq \lambda$$

or $$2 \leq \lambda$$

So we have a feasible solution with $\lambda = 2$.

Verifying the Dual



Verifying the Dual

$$\min \lambda$$

$$
\begin{array}{llll}
2x_1 & & +d & \le \lambda d \\
2x_1 + 2x_2 & & +2d & \le 8x_1 + \lambda d \\
2x_1 + 2x_2 + 2x_3 & & +3d & \le 8x_2 + \lambda d \\
\vdots & \ddots & \vdots & \vdots \\
2x_1 + 2x_2 + 2x_3 & +2x_{n+1} & +(n+1)d & \le 8x_n + \lambda d \\
\vdots & \ddots & \vdots & \vdots \\
 & & x_i & \ge 0
\end{array}
$$

Consider infinite linear combination of with the dual multipliers:

The resulting coefficient of x_n is

$$\sum_{i=n}^{\infty} \frac{2}{2^i} - \frac{8}{2^{n+1}} = 0$$

The resulting coefficient of λd is

$$\sum_{i=1}^{\infty} \frac{1}{2^i} = 1$$

Verifying the Dual

$$\min \lambda$$

$$
\begin{aligned}
2x_1 \quad & + d \le \lambda d \\
2x_1 + 2x_2 \quad & + 2d \le 8x_1 + \lambda d \\
2x_1 + 2x_2 + 2x_3 \quad & + 3d \le 8x_2 + \lambda d \\
\vdots \qquad & \qquad \vdots \\
2x_1 + 2x_2 + 2x_3 \;\cdots\; + 2x_{n+1} \quad & + (n+1)d \le 8x_n + \lambda d \\
\vdots \qquad & \qquad \vdots \\
x_i & \ge 0
\end{aligned}
$$

Consider infinite linear combination of with the dual multipliers:

The resulting coefficient of x_n is

$$\sum_{i=n}^{\infty} \frac{2}{2^i} - \frac{8}{2^{n+1}} = 0$$

The resulting coefficient of λd is

$$\sum_{i=1}^{\infty} \frac{1}{2^i} = 1$$

This leaves the inequality

$$\sum_{i=1}^{\infty} i \left(\tfrac{1}{2}\right)^i d \le \lambda d$$

Using $\sum_{i=1}^{\infty} i x^i = \frac{x}{(1-x)^2}$, this implies

$$2 \le \lambda,$$

so we have an explicit lower bound.

More General Problem

Cow-Path Problem with Turn Cost

Scenario: m rays from the origin.

Turn cost on a ray: d_1

Turn cost at the origin: d_2

Total turn cost for changing from one ray to another: $d = d_1 + d_2$

Known: Asymptotic competitive ratio for $d = 0$ is

$$1 + \frac{2m^m}{(m-1)^{m-1}} =: 1 + M$$

Constraints

REWRITE CONSTRAINTS:

$$2\sum_{i=1}^{n+m-1} x_i + (n+m-1)\,d \le M x_n + \lambda d$$

AGAIN:

- Infinite LP for determining λ
- Run experiments for fixed m

Solving the Problem

SOLUTION OF THE PROBLEM

Here described: $m = 3$

$$\lambda_{1000} = 3.743996$$

$$x_1^{(1000)} = 0.2492495$$

$$x_2^{(1000)} = 0.6227485$$

$$x_3^{(1000)} = 1.182434$$

$$x_4^{(1000)} = 2.021118$$

$$x_5^{(1000)} = 3.277878$$

Solving the Problem

Solution of the Problem

Here described: $m = 3$

$$\lambda_{1000} = 3.743996$$
$$x_1^{(1000)} = 0.2492495$$
$$x_2^{(1000)} = 0.6227485$$
$$x_3^{(1000)} = 1.182434$$
$$x_4^{(1000)} = 2.021118$$
$$x_5^{(1000)} = 3.277878$$

After adjusting for logarithmic convergence:

$$\left.\begin{aligned} \lambda &= 3.75 = \tfrac{15}{4} \\ x_1 &= 0.25 = \tfrac{1}{4} \\ x_2 &= 0.625 = \tfrac{5}{8} \end{aligned}\right\} \text{educated guesses}$$

Assuming all constraints are tight, we get a recursion for x_n, yielding:

$$x_3 = \tfrac{19}{16} = 1.1875 \quad \checkmark$$
$$x_4 = \tfrac{65}{32} = 2.03125 \quad \checkmark$$
$$x_5 = \tfrac{211}{64} = 3.296875 \quad \checkmark$$

Solution II

Solution for m=3 (Cont.)

Using the structure of the recursion, we conclude

$$x_n = \frac{d}{2}\left(\left(\frac{3}{2}\right)^n - 1\right)$$

Not hard to check:

Together with $\lambda = \frac{15}{4}$, this satisfies all constraints with equality.

Dual Variables

$$c_2^{(1000)} = 0.445339$$
$$c_3^{(1000)} = 0.1481481$$
$$c_4^{(1000)} = 0.1481481$$
$$c_5^{(1000)} = 0.08217275$$
$$c_6^{(1000)} = 0.06022488$$
$$c_7^{(1000)} = 0.038277$$
$$c_8^{(1000)} = 0.02610326$$

Using (*), we get the recursive condition

$$c_n = \frac{27}{4}\left(c_{n+2} - c_{n+3}\right)$$

or

$$c_{n+3} = \frac{27}{4} c_{n+2} - c_n .$$

Some values:

$$c_5 = \frac{60}{3^6} = 0.0823045 \quad ✓$$
$$c_6 = \frac{132}{3^7} = 0.0603566 \quad ✓$$
$$c_7 = \frac{252}{3^8} = 0.0384087 \quad ✓$$
$$c_8 = \frac{516}{3^9} = 0.0262155 \quad ✓$$

Dual Routing

Explicit formula after solving recursion:

$$c_j = \frac{2^{j+1} + (-1)^j 4}{3^{j+1}}$$

Dual Routing

Dual Routing

VERIFYING THE DUAL

Consider the infinite linear combination of all constraints, using the computed c_j.

- By assumption, we have

$$\sum_{i=2}^{\infty} c_i = 1$$

so the coefficient of $2d$ is 1.

- By recursion, all coefficients of x_n cancel.

Dual Routing

- This leaves

$$\sum_{i=2}^{\infty} i c_i \leq \lambda$$

Using the explicit values of c_i and $\sum_{i=1}^{\infty} i x^i = \frac{x}{(1-x)^2}$,

we get

$$\lambda \geq \sum_{i=2}^{\infty} i c_i = \frac{2}{3}\sum_{i=1}^{\infty} i \left(\frac{2}{3}\right)^i + \frac{4}{3}\sum_{i=1}^{\infty} i \left(-\frac{1}{3}\right)^i$$

$$= \frac{2}{3}\,\frac{\frac{2}{3}}{\left(1-\frac{2}{3}\right)^2} + \frac{4}{3}\,\frac{-\frac{1}{3}}{\left(1+\frac{1}{3}\right)^2}$$

$$= 4 - \frac{1}{4} = \frac{15}{4} = 3.75$$

ELSEVIER

Available online at www.sciencedirect.com

ScienceDirect

Theoretical Computer Science 361 (2006) 342–355

Theoretical Computer Science

www.elsevier.com/locate/tcs

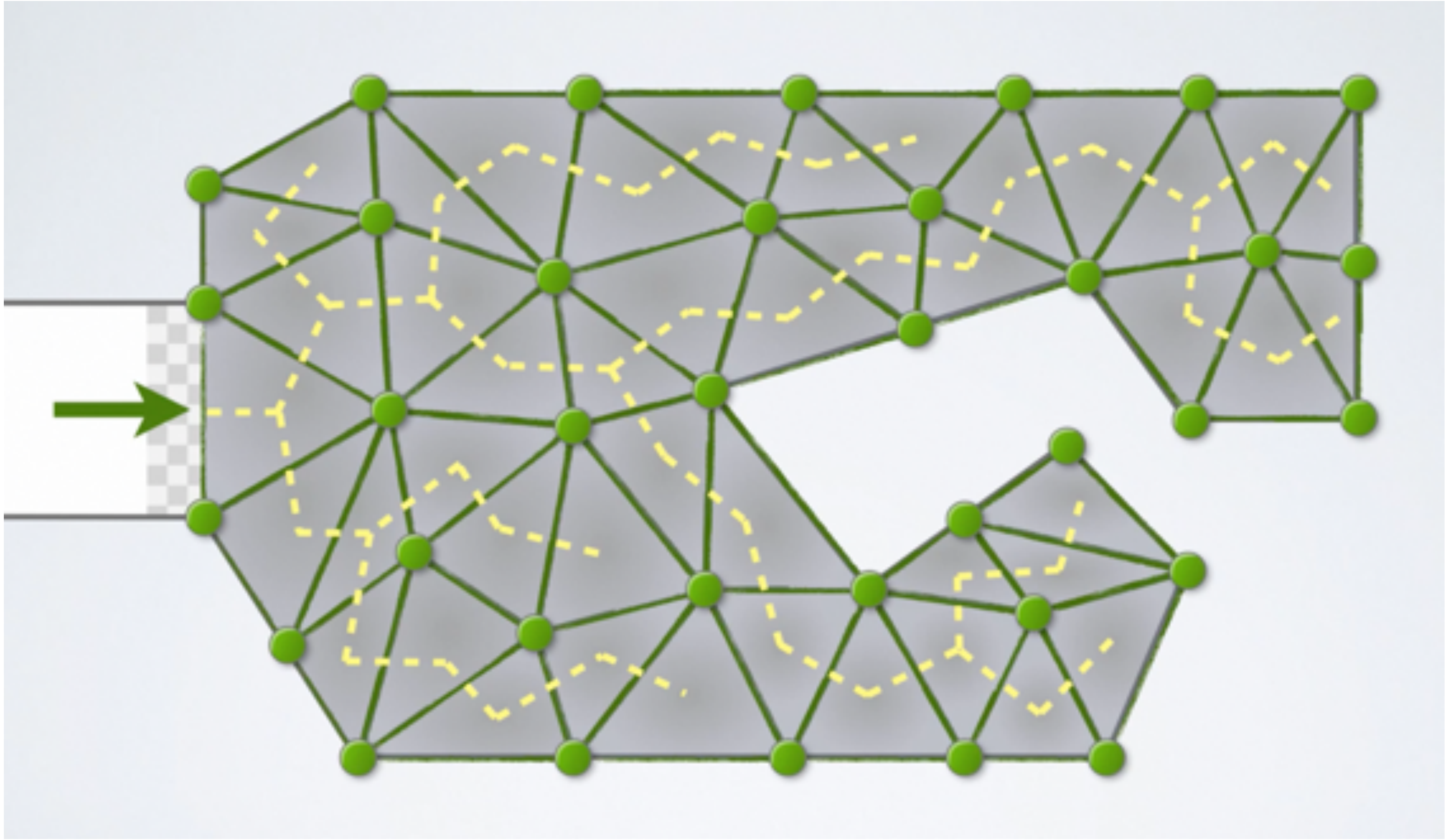
Online searching with turn cost

Erik D. Demaine[a], Sándor P. Fekete[b,*], Shmuel Gal[c]

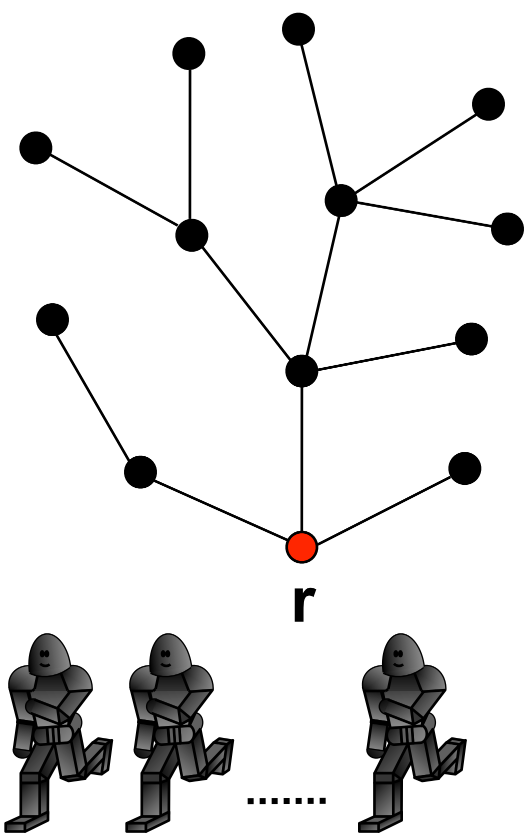
[a] Computer Science and Artificial Intelligence Laboratory, MIT, Cambridge MA, USA
[b] Department of Mathematical Optimization, Braunschweig University of Technology, Braunschweig, Germany
[c] Department of Statistics, University of Haifa, Haifa, Israel

Part 2: Several Robots

Collective Tree Exploration



Tree Exploration



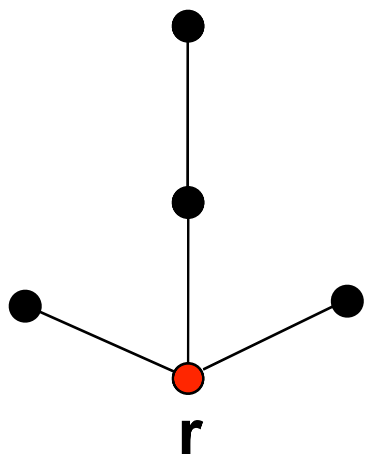
Given:
Unknown tree T, root r
k robots, initially located at r

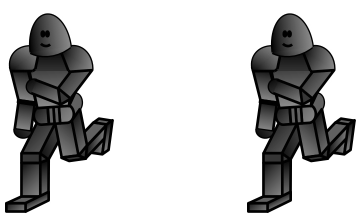
Task:
Explore T and return to origin

Objective:
Minimize maximum workload

Previous Work

$$\frac{ALG}{OPT} = \frac{6}{4} = \frac{3}{2}$$

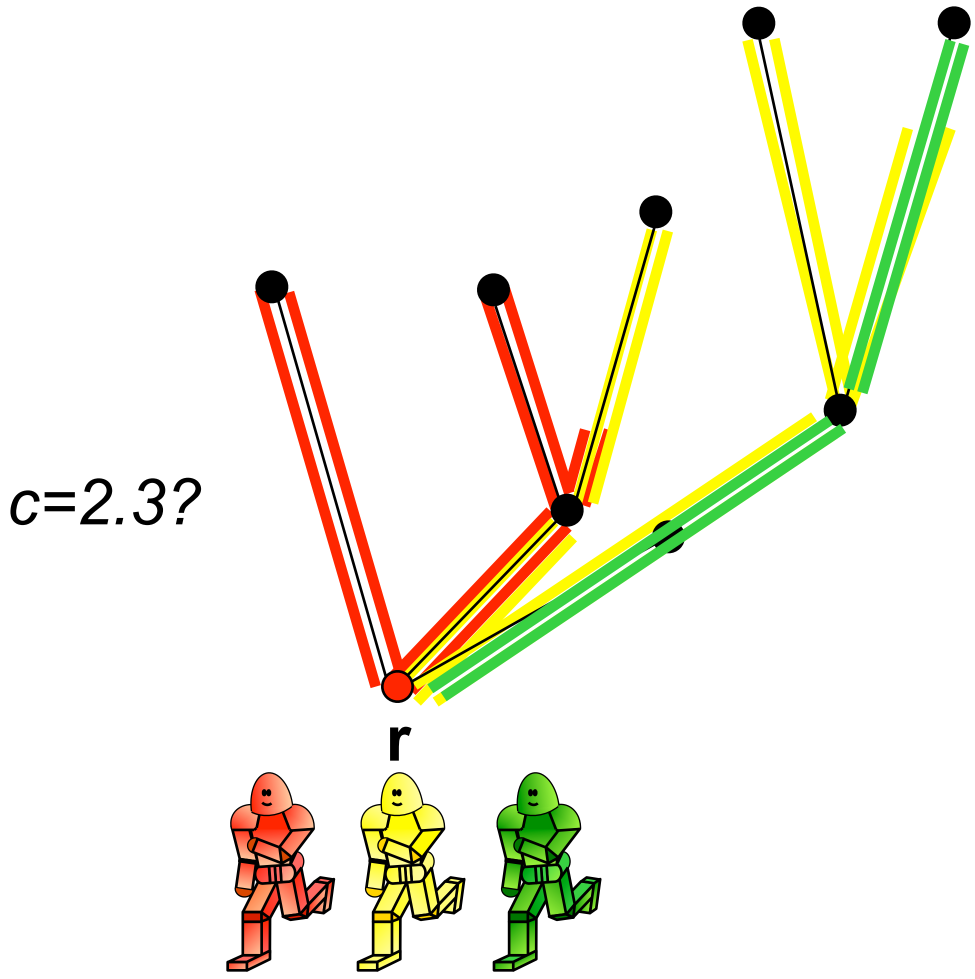




Dynia et al. (2006):

- Lower bound of 3/2 on competitive factor
- An appropriate greedy algorithm achieves competitive factor of 8

A New Strategy for General Trees



- Lower bounds on actual OPT:
  - Known MAX distance
  - AVG of known total distance
- Strategy MAX+AVG:
  - Choose some *c*.
  - Robots take turns, one at a time.
  - Keep track of MAX and AVG.
  - Travel c times lower bound.
- Factor *c* is achievable, if we can keep going - so if we can travel arbitrarily far.
- Observations:
  - Duplicated distance DUP is bounded by MAX.
  - In worst case, MAX=AVG=DUP.
  - This yields a recursion for distances traveled.

Recursion

D_i : total distance traveled by a robot after iteration i

d_i : new distance traveled by a robot in iteration i

New total:

$$\underbrace{d_i}_{\text{new}} + \underbrace{D_{i-k}}_{\text{old total}} + \underbrace{\frac{D_{i-1}}{c}}_{\text{duplicated}} = c\left(\underbrace{\frac{D_{i-1}}{c}}_{\text{old average}} + \underbrace{\frac{d_i}{k}}_{\text{added to average}}\right)$$

Rearrange:

$$D_i = \left(\frac{k-1}{k-c}\right) D_{i-1} - \left(\frac{c}{k-c}\right) D_{i-k}$$

Analysis

▶ **Lemma 1.** *For any fixed k, Strategy MAX+AVG is c_k-competitive, where c_k satisfies $c_k = \frac{kx_k^k-(k-1)x_k^{k-1}}{x_k^k-1}$, with $x_k > 1$ being a zero of the function $f(x) = (k-1)x^k - k^2x + k^2 - 2k + 1$.*

$$x_k^k - \frac{k-1}{(k-c_k)}x_k^{k-1} + \frac{c_k}{k-c_k} = 0$$

$$x_k^{k-1} \geq \frac{c_k}{c_k - 1}$$

$$x_k > 1$$

Analysis

▶ **Lemma 1.** *For any fixed k, Strategy MAX+AVG is c_k-competitive, where c_k satisfies $c_k = \frac{k x_k^k - (k-1) x_k^{k-1}}{x_k^k - 1}$, with $x_k > 1$ being a zero of the function $f(x) = (k-1)x^k - k^2 x + k^2 - 2k + 1$.*

$$x_k^k - \frac{k-1}{(k-c_k)} x_k^{k-1} + \frac{c_k}{k-c_k} = 0$$

$$x_k^{k-1} = \frac{c_k}{c_k - 1}$$

$$x_k > 1$$

$$\left(1 + \frac{1}{c_k - 1}\right)^{\frac{1}{k-1}} = 1.$$

Analysis

▶ **Lemma 1.** *For any fixed k, Strategy MAX+AVG is c_k-competitive, where c_k satisfies $c_k = \frac{kx_k^k-(k-1)x_k^{k-1}}{x_k^k-1}$, with $x_k > 1$ being a zero of the function $f(x) = (k-1)x^k - k^2x + k^2 - 2k + 1$.*

$$x_k^k - \frac{k-1}{(k-c_k)}x_k^{k-1} + \frac{c_k}{k-c_k} = 0$$

$$x_k^{k-1} \geq \frac{c_k}{c_k - 1}$$

$$x_k > 1$$

Analysis

▶ **Lemma 1.** *For any fixed k, Strategy MAX+AVG is c_k-competitive, where c_k satisfies $c_k = \frac{kx_k^k-(k-1)x_k^{k-1}}{x_k^k-1}$, with $x_k > 1$ being a zero of the function $f(x) = (k-1)x^k - k^2x + k^2 - 2k + 1$.*

$$x_k^k - \frac{k-1}{(k-c_k)}x_k^{k-1} + \frac{c_k}{k-c_k} = 0$$

$$(k-c_k)x_k^k - (k-1)x_k^{k-1} + c_k = 0$$

$$c_k = \frac{kx_k^k - (k-1)x_k^{k-1}}{x_k^k - 1}.$$

Derivative:

$$\frac{x_k^{k-2}\left((k-1)x_k^k - k^2x_k + k^2 - 2k + 1\right)}{\left(x_k^k - 1\right)^2}$$

Analysis

▶ **Lemma 1.** *For any fixed k, Strategy MAX+AVG is c_k-competitive, where c_k satisfies* $c_k = \frac{kx_k^k-(k-1)x_k^{k-1}}{x_k^k-1}$, *with $x_k > 1$ being a zero of the function* $f(x) = (k-1)x^k - k^2x + k^2 - 2k + 1$.

k	c_k
2	
3	
4	
5	
6	
7	
8	
9	
10	
20	
40	
100	
1,000	
10,000	
100,000	
1,000,000	

Analysis

▶ **Lemma 1.** *For any fixed k, Strategy MAX+AVG is c_k-competitive, where c_k satisfies* $c_k = \frac{k x_k^k - (k-1) x_k^{k-1}}{x_k^k - 1}$, *with* $x_k > 1$ *being a zero of the function* $f(x) = (k-1)x^k - k^2 x + k^2 - 2k + 1$.

k	c_k
2	1.86603...
3	
4	
5	
6	
7	
8	
9	
10	
20	
40	
100	
1,000	
10,000	
100,000	
1,000,000	

Analysis

▶ **Lemma 1.** *For any fixed k, Strategy MAX+AVG is c_k-competitive, where c_k satisfies* $c_k = \frac{kx_k^k - (k-1)x_k^{k-1}}{x_k^k - 1}$, *with $x_k > 1$ being a zero of the function* $f(x) = (k-1)x^k - k^2x + k^2 - 2k + 1$.

k	c_k
2	1.86603...
3	2.27883...
4	
5	
6	
7	
8	
9	
10	
20	
40	
100	
1,000	
10,000	
100,000	
1,000,000	

Analysis

▶ **Lemma 1.** *For any fixed k, Strategy MAX+AVG is c_k-competitive, where c_k satisfies* $c_k = \frac{kx_k^k-(k-1)x_k^{k-1}}{x_k^k-1}$, *with $x_k > 1$ being a zero of the function* $f(x) = (k-1)x^k - k^2x + k^2 - 2k + 1$.

k	c_k
2	1.86603...
3	2.27883...
4	2.49221...
5	
6	
7	
8	
9	
10	
20	
40	
100	
1,000	
10,000	
100,000	
1,000,000	

Analysis

▶ **Lemma 1.** *For any fixed k, Strategy MAX+AVG is c_k-competitive, where c_k satisfies $c_k = \frac{kx_k^k - (k-1)x_k^{k-1}}{x_k^k - 1}$, with $x_k > 1$ being a zero of the function $f(x) = (k-1)x^k - k^2x + k^2 - 2k + 1$.*

k	c_k
2	1.86603...
3	2.27883...
4	2.49221...
5	2.62163...
6	
7	
8	
9	
10	
20	
40	
100	
1,000	
10,000	
100,000	
1,000,000	

Analysis

▶ **Lemma 1.** *For any fixed k, Strategy MAX+AVG is c_k-competitive, where c_k satisfies $c_k = \frac{k x_k^k - (k-1) x_k^{k-1}}{x_k^k - 1}$, with $x_k > 1$ being a zero of the function $f(x) = (k-1)x^k - k^2 x + k^2 - 2k + 1$.*

k	c_k
2	1.86603...
3	2.27883...
4	2.49221...
5	2.62163...
6	2.70837...
7	
8	
9	
10	
20	
40	
100	
1,000	
10,000	
100,000	
1,000,000	

Analysis

▶ **Lemma 1.** *For any fixed k, Strategy MAX+AVG is c_k-competitive, where c_k satisfies* $c_k = \frac{kx_k^k - (k-1)x_k^{k-1}}{x_k^k - 1}$*, with $x_k > 1$ being a zero of the function* $f(x) = (k-1)x^k - k^2x + k^2 - 2k + 1$.

k	c_k
2	1.86603...
3	2.27883...
4	2.49221...
5	2.62163...
6	2.70837...
7	2.77053...
8	
9	
10	
20	
40	
100	
1,000	
10,000	
100,000	
1,000,000	

Analysis

▶ **Lemma 1.** *For any fixed k, Strategy MAX+AVG is c_k-competitive, where c_k satisfies* $c_k = \frac{kx_k^k - (k-1)x_k^{k-1}}{x_k^k - 1}$, *with $x_k > 1$ being a zero of the function* $f(x) = (k-1)x^k - k^2x + k^2 - 2k + 1$.

k	c_k
2	1.86603...
3	2.27883...
4	2.49221...
5	2.62163...
6	2.70837...
7	2.77053...
8	2.81724...
9	
10	
20	
40	
100	
1,000	
10,000	
100,000	
1,000,000	

Analysis

▶ **Lemma 1.** *For any fixed k, Strategy MAX+AVG is c_k-competitive, where c_k satisfies* $c_k = \frac{kx_k^k - (k-1)x_k^{k-1}}{x_k^k - 1}$, *with $x_k > 1$ being a zero of the function* $f(x) = (k-1)x^k - k^2x + k^2 - 2k + 1$.

k	c_k
2	1.86603...
3	2.27883...
4	2.49221...
5	2.62163...
6	2.70837...
7	2.77053...
8	2.81724...
9	2.85363...
10	
20	
40	
100	
1,000	
10,000	
100,000	
1,000,000	

Analysis

▶ **Lemma 1.** *For any fixed* k*, Strategy MAX+AVG is* c_k*-competitive, where* c_k *satisfies* $c_k = \frac{k x_k^k - (k-1) x_k^{k-1}}{x_k^k - 1}$*, with* $x_k > 1$ *being a zero of the function* $f(x) = (k-1)x^k - k^2 x + k^2 - 2k + 1$.

k	c_k
2	1.86603...
3	2.27883...
4	2.49221...
5	2.62163...
6	2.70837...
7	2.77053...
8	2.81724...
9	2.85363...
10	2.88277...
20	
40	
100	
1,000	
10,000	
100,000	
1,000,000	

Analysis

▶ **Lemma 1.** *For any fixed k, Strategy MAX+AVG is c_k-competitive, where c_k satisfies* $c_k = \frac{kx_k^k - (k-1)x_k^{k-1}}{x_k^k - 1}$, *with* $x_k > 1$ *being a zero of the function* $f(x) = (k-1)x^k - k^2x + k^2 - 2k + 1$.

k	c_k
2	1.86603...
3	2.27883...
4	2.49221...
5	2.62163...
6	2.70837...
7	2.77053...
8	2.81724...
9	2.85363...
10	2.88277...
20	3.01425...
40	
100	
1,000	
10,000	
100,000	
1,000,000	

Analysis

▶ **Lemma 1.** *For any fixed k, Strategy MAX+AVG is c_k-competitive, where c_k satisfies $c_k = \frac{kx_k^k - (k-1)x_k^{k-1}}{x_k^k - 1}$, with $x_k > 1$ being a zero of the function $f(x) = (k-1)x^k - k^2x + k^2 - 2k + 1$.*

k	c_k
2	1.86603...
3	2.27883...
4	2.49221...
5	2.62163...
6	2.70837...
7	2.77053...
8	2.81724...
9	2.85363...
10	2.88277...
20	3.01425...
40	3.08016...
100	
1,000	
10,000	
100,000	
1,000,000	

Analysis

▶ **Lemma 1.** *For any fixed k, Strategy MAX+AVG is c_k-competitive, where c_k satisfies* $c_k = \frac{kx_k^k-(k-1)x_k^{k-1}}{x_k^k-1}$, *with $x_k > 1$ being a zero of the function* $f(x) = (k-1)x^k - k^2x + k^2 - 2k + 1$.

k	c_k
2	1.86603...
3	2.27883...
4	2.49221...
5	2.62163...
6	2.70837...
7	2.77053...
8	2.81724...
9	2.85363...
10	2.88277...
20	3.01425...
40	3.08016...
100	3.11977...
1,000	
10,000	
100,000	
1,000,000	

Analysis

▶ **Lemma 1.** *For any fixed k, Strategy MAX+AVG is c_k-competitive, where c_k satisfies* $c_k = \frac{kx_k^k - (k-1)x_k^{k-1}}{x_k^k - 1}$, *with $x_k > 1$ being a zero of the function* $f(x) = (k-1)x^k - k^2x + k^2 - 2k + 1$.

k	c_k
2	1.86603...
3	2.27883...
4	2.49221...
5	2.62163...
6	2.70837...
7	2.77053...
8	2.81724...
9	2.85363...
10	2.88277...
20	3.01425...
40	3.08016...
100	3.11977...
1,000	3.14
10,000	
100,000	
1,000,000	

Analysis

▶ **Lemma 1.** *For any fixed k, Strategy MAX+AVG is c_k-competitive, where c_k satisfies* $c_k = \frac{k x_k^k - (k-1) x_k^{k-1}}{x_k^k - 1}$, *with $x_k > 1$ being a zero of the function* $f(x) = (k-1)x^k - k^2 x + k^2 - 2k + 1$.

k	c_k
2	1.86603...
3	2.27883...
4	2.49221...
5	2.62163...
6	2.70837...
7	2.77053...
8	2.81724...
9	2.85363...
10	2.88277...
20	3.01425...
40	3.08016...
100	3.11977...
1,000	3.14355...
10,000	
100,000	
1,000,000	

Analysis

▶ **Lemma 1.** *For any fixed* k*, Strategy MAX+AVG is* c_k*-competitive, where* c_k *satisfies* $c_k = \frac{kx_k^k-(k-1)x_k^{k-1}}{x_k^k-1}$*, with* $x_k > 1$ *being a zero of the function* $f(x) = (k-1)x^k - k^2x + k^2 - 2k + 1$.

k	c_k
2	1.86603...
3	2.27883...
4	2.49221...
5	2.62163...
6	2.70837...
7	2.77053...
8	2.81724...
9	2.85363...
10	2.88277...
20	3.01425...
40	3.08016...
100	3.11977...
1,000	3.14355...
10,000	3.14592...
100,000	
1,000,000	

Analysis

▶ **Lemma 1.** *For any fixed k, Strategy MAX+AVG is c_k-competitive, where c_k satisfies* $c_k = \frac{k x_k^k - (k-1) x_k^{k-1}}{x_k^k - 1}$, *with* $x_k > 1$ *being a zero of the function* $f(x) = (k-1)x^k - k^2 x + k^2 - 2k + 1$.

k	c_k
2	1.86603...
3	2.27883...
4	2.49221...
5	2.62163...
6	2.70837...
7	2.77053...
8	2.81724...
9	2.85363...
10	2.88277...
20	3.01425...
40	3.08016...
100	3.11977...
1,000	3.14355...
10,000	3.14592...
100,000	3.14612...
1,000,000	

Analysis

▶ **Lemma 1.** *For any fixed k, Strategy MAX+AVG is c_k-competitive, where c_k satisfies* $c_k = \frac{kx_k^k-(k-1)x_k^{k-1}}{x_k^k-1}$, *with $x_k > 1$ being a zero of the function $f(x) = (k-1)x^k - k^2x + k^2 - 2k + 1$.*

k	c_k
2	1.86603...
3	2.27883...
4	2.49221...
5	2.62163...
6	2.70837...
7	2.77053...
8	2.81724...
9	2.85363...
10	2.88277...
20	3.01425...
40	3.08016...
100	3.11977...
1,000	3.14355...
10,000	3.14592...
100,000	3.14612...
1,000,000	3.14619...

▶ **Theorem 2.** *Strategy MAX+AVG is c_k-competitive, for the values shown in Table 1. Moreover, these values are tight.*

Analysis

$$c_k = \frac{kx_k^k - (k-1)x_k^{k-1}}{x_k^k - 1}$$

Analysis

$$c_k = \frac{k x_k^k - (k-1) x_k^{k-1}}{x_k^k - 1}$$

$$x_k = \left(1 + \frac{z_k}{k}\right)$$

Analysis

$$c_k = \frac{k x_k^k - (k-1) x_k^{k-1}}{x_k^k - 1}$$

$$x_k = \left(1 + \frac{z_k}{k}\right)$$

$$c_k = \frac{k(1+\frac{z_k}{k}) - (k-1)}{(1+\frac{z_k}{k}) - \frac{1}{(1+\frac{z_k}{k})^{k-1}}} = \frac{1+z_k}{(1+\frac{z_k}{k}) - \frac{1}{(1+\frac{z_k}{k})^{k-1}}}$$

Analysis

$$c_k = \frac{k(1+\frac{z_k}{k}) - (k-1)}{(1+\frac{z_k}{k}) - \frac{1}{(1+\frac{z_k}{k})^{k-1}}} = \frac{1+z_k}{(1+\frac{z_k}{k}) - \frac{1}{(1+\frac{z_k}{k})^{k-1}}}$$

$$\lim_{k\to\inf} \frac{1+z_k}{(1+\frac{z_k}{k}) - \frac{1}{(1+\frac{z_k}{k})^{k-1}}} = \frac{1+z}{1-e^{-z}}$$

Derivative:

$$\frac{e^z(-z+e^z-2)}{(e^z-1)^2}$$

Zero of derivative:

$$e^z = z+2$$

$$c = W_{-1}(-\tfrac{1}{e^2}) = 3.146193220582\ldots$$

Analysis

▶ **Theorem 3.** *Algorithm MAX+AVG is c-competitive for all k, where c is the solution of the equation* $e^c = c + 2$. *This is the value* $W_{-1}(-\frac{1}{e^2}) = 3.146193220582\ldots$, *where* W_{-1} *is the lower branch of Lambert's W-function. Moreover, this is tight: For any* $c' < c$, *MAX+AVG is not* c'*-competitive for large enough k.*

$$\frac{e^z(-z+e^z-2)}{(e^z-1)^2}$$

$$e^z = z + 2$$

$$c = W_{-1}(-\tfrac{1}{e^2}) = 3.146193220582\ldots$$

Part 3: Robot Swarms

Part 3.1:
Online Triangulation

Video!


Triangulating Unknown Environments
using Robot Swarms
Aaron Becker
James McLurkin
SeoungKyou Lee
RICE
Sándor P. Fekete
Alexander Kröller
Christiane Schmidt
Technische
Universität
Braunschweig

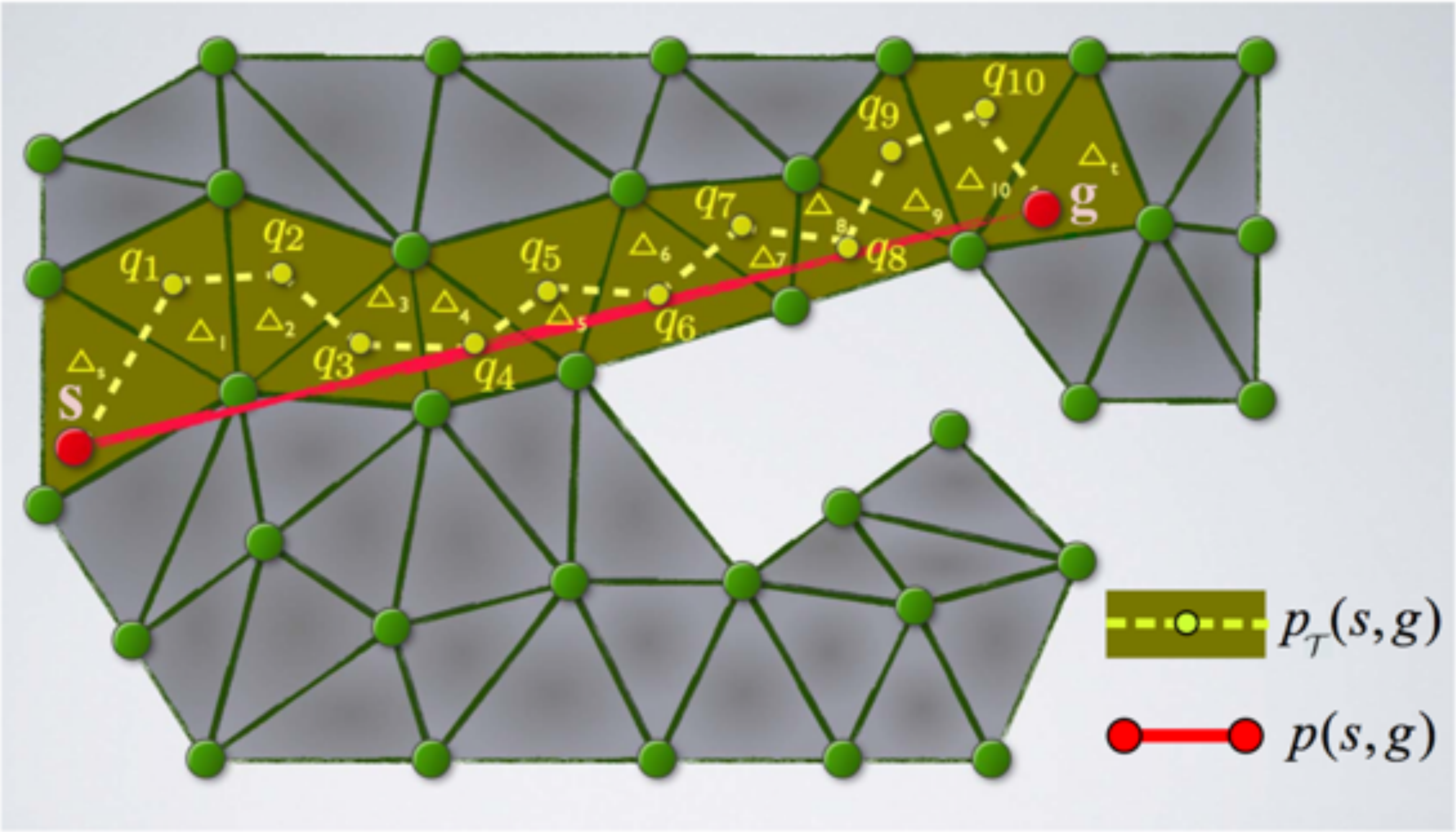
Video!

conference

S.P. Fekete, A. Kröller, L.S. Kyou, J. McLurkin, C. Schmidt:
Triangulating Unknown Environments Using Robot Swarms,
Video and abstract. In: Proceedings of the 29th Annual ACM Symposium on Computational Geometry (SoCG 2013), 345-346.

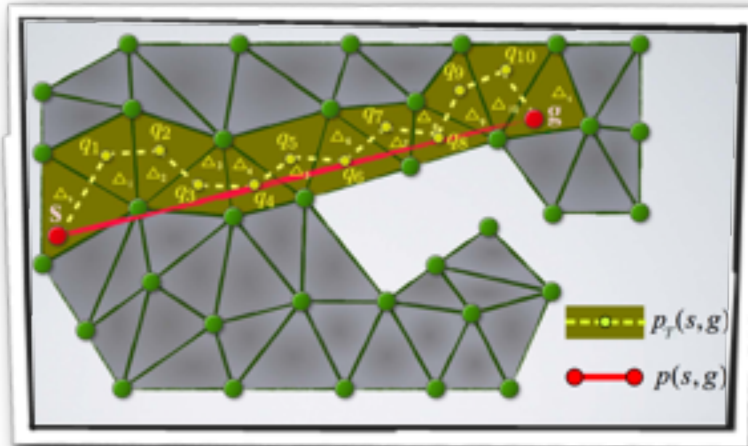
Part 3.2:
Local Routing

Dual Routing



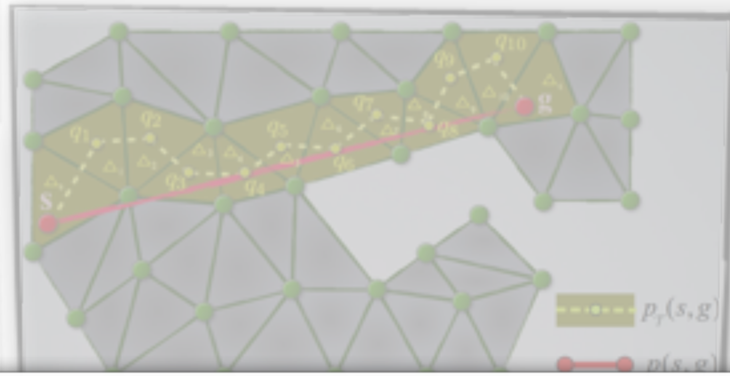
Note: The dual graph is stored implicitly in *primal* vertices!

Dual Routing



Theorem 3.3: Consider a (ρ, α)-fat triangulation $\mathcal{T}$ of a planar region $\mathcal{R}$, with vertex set V, maximum and minimum edge length r_{max} and r_{min}, respectively. Let s, g be points in $\mathcal{R}$ that are separated by at least one triangle, i.e., the triangles Δ_s, Δ_g in $\mathcal{T}$ that contain s and g do not share a vertex. Let $p(s,g)$ be a shortest polygonal path in $\mathcal{R}$ that connects s with g, and let $d_p(s,g)$ be its length. Let $p_{\mathcal{T}}(s,g)$ be a $\mathcal{T}$-greedy path between s and g, of length $d_{p_{\mathcal{T}}}(s,g)$. Then $d_{p_{\mathcal{T}}}(s,g) \leq c \cdot d_p(s,g) + 2$, for $c = \lfloor \frac{2\pi}{\alpha} \rfloor \frac{\rho}{\sin(\alpha/2)}$, and $d_{p_{\mathcal{T}}}(s,g) \leq c' \cdot d_p(s,g)$, for $c' = \lfloor \frac{6\pi}{\alpha} \rfloor \frac{\rho}{\sin(\alpha/2)}$.

Dual Routing



conference

S. K. Lee, A. Becker, S.P. Fekete, A. Kröller, J. McLurkin:

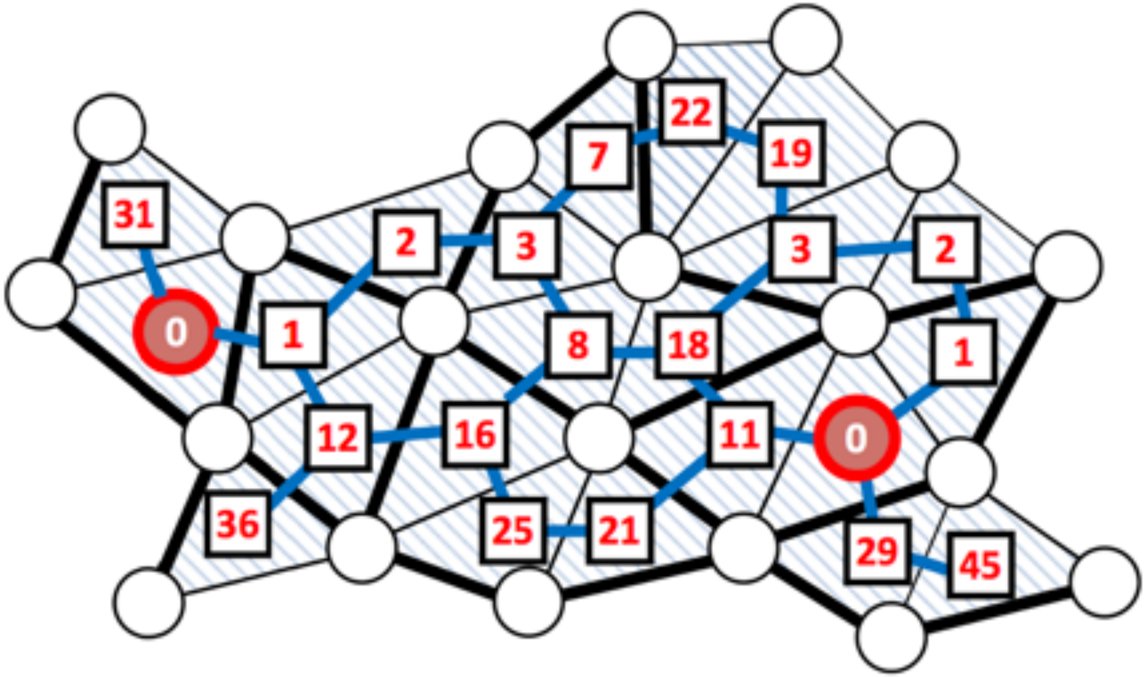
Exploration via Structured Triangulation by a Multi-Robot System with Bearing-Only Low-Resolution Sensors,

NEW To appear in: 2014 IEEE International Conference on Robotics and Automation (ICRA 2014)

in $\mathcal{R}$ that are separated by at least one triangle, i.e., the triangles Δ_s, Δ_g in $\mathcal{T}$ that contain s and g do not share a vertex. Let $p(s,g)$ be a shortest polygonal path in $\mathcal{R}$ that connects s with g, and let $d_p(s,g)$ be its length. Let $p_{\mathcal{T}}(s,g)$ be a $\mathcal{T}$-greedy path between s and g, of length $d_{p_{\mathcal{T}}}(s,g)$. Then $d_{p_{\mathcal{T}}}(s,g) \leq c \cdot d_p(s,g) + 2$, for $c = \lfloor \frac{2\pi}{\alpha} \rfloor \frac{\rho}{\sin(\alpha/2)}$, and $d_{p_{\mathcal{T}}}(s,g) \leq c' \cdot d_p(s,g)$, for $c' = \lfloor \frac{6\pi}{\alpha} \rfloor \frac{\rho}{\sin(\alpha/2)}$.

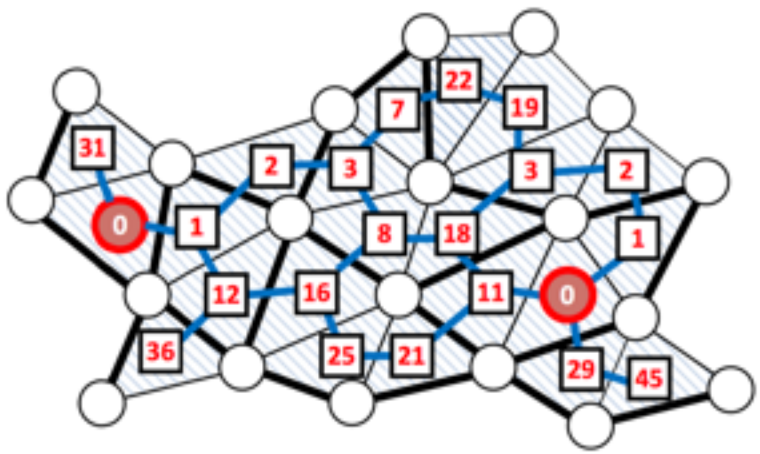
Part 3.3: Local Patrolling Policies

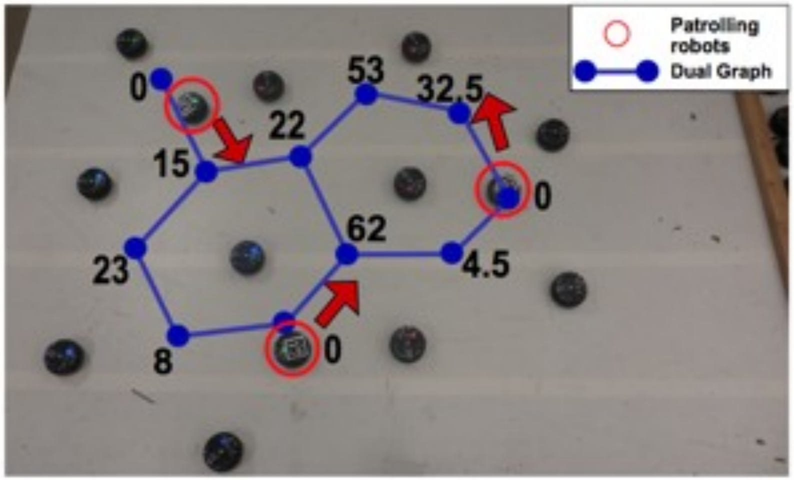
Time Stamps in the Dual Graph



Numbers: Time of last visit

Least Recently Visited



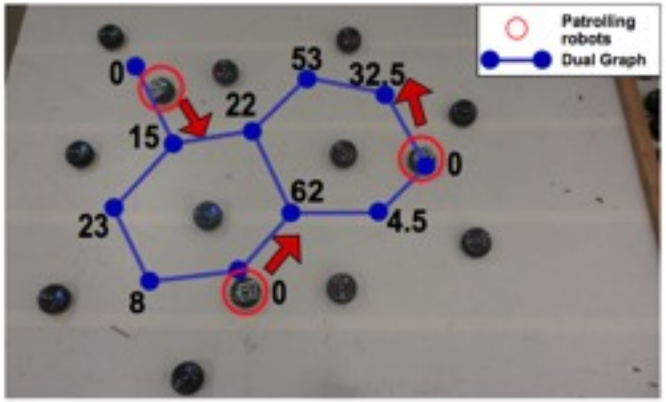


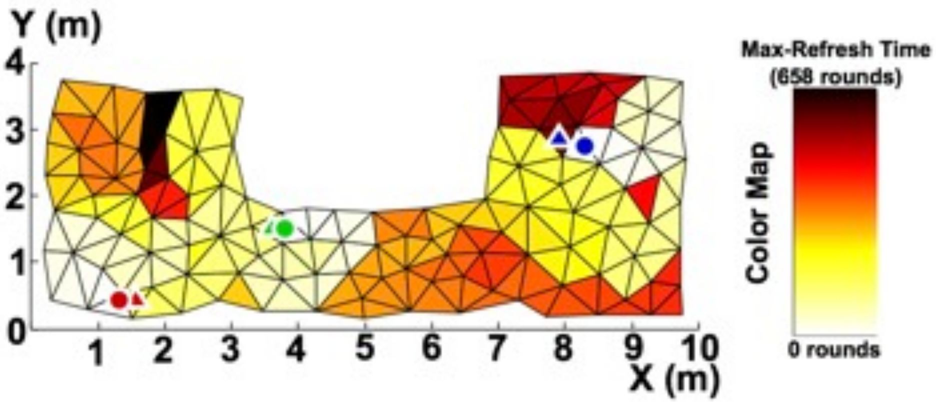
Least Recently Visited (LRV):
Move to vertex with oldest time stamp

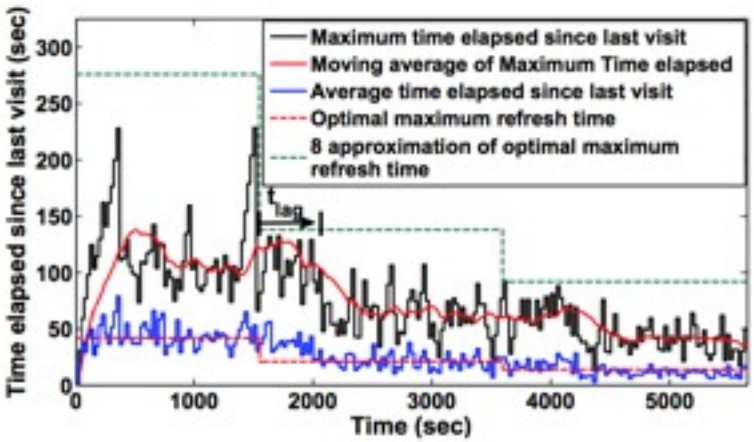
Good news: LRV achieves full coverage.

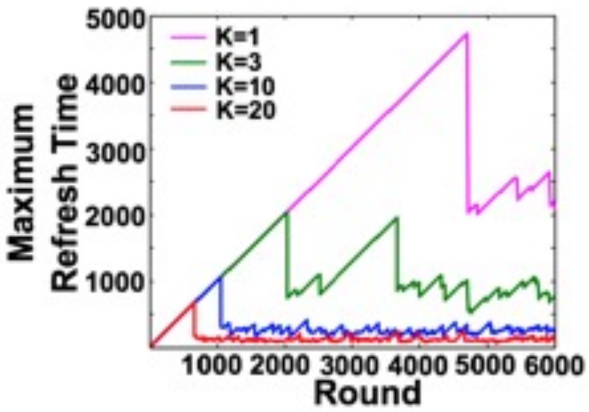
Bad news: The coverage time of LRV can be exponentially large.

LRV: Experimental Results



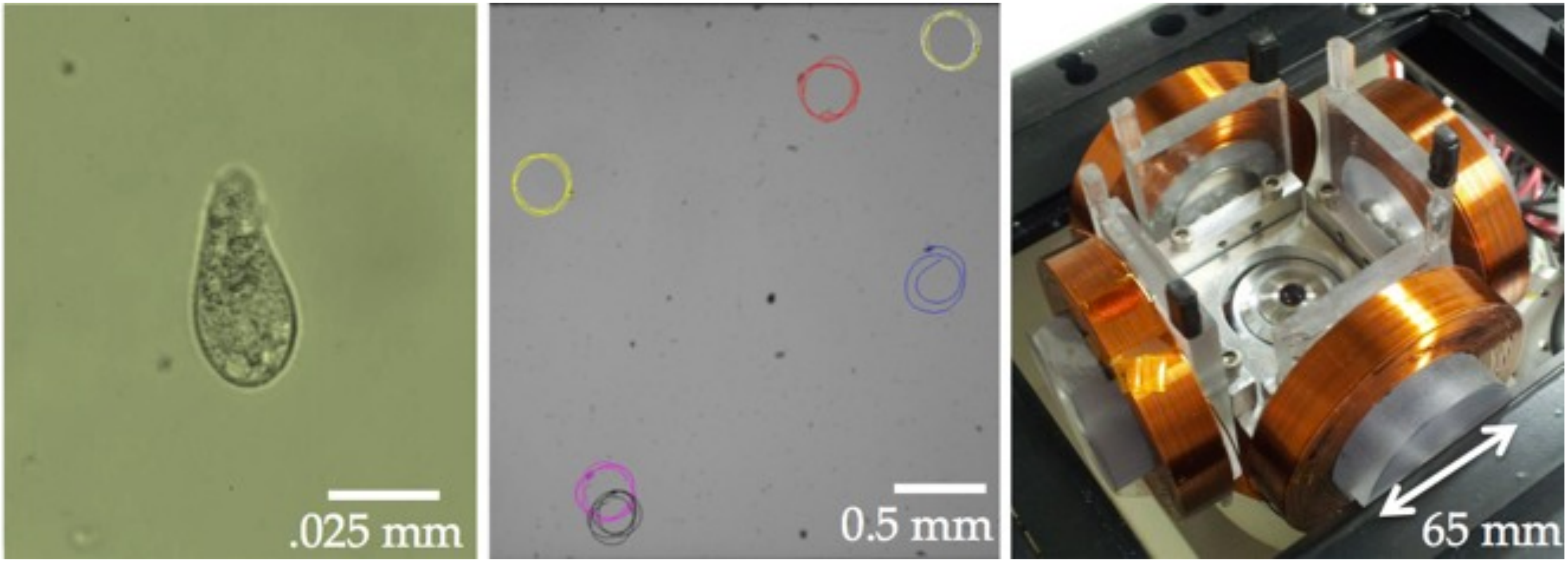






Part 4:
Controlling Massive Particle Swarms

Moving Small Objects



Tetrahymena pyriformis

This Part

- Massive particle swarms
- Global control, not individual motion
  - *We show hardness for given, external obstacles*
  - *We establish positive results for designed, additional obstacles*
- Work in progress, combining theory and practice

This Part

- Massive particle swarms

conference

A. Becker, E.D. Demaine, S.P. Fekete, G. Habibi, J. McLurkin:
Reconfiguring Massive Particle Swarms with Limited, Global Control,
NEW In: ALGOSENSORS 2013, pp. 51-66, Springer LNCS 8343, 2014.

- *We establish positive results for*

conference

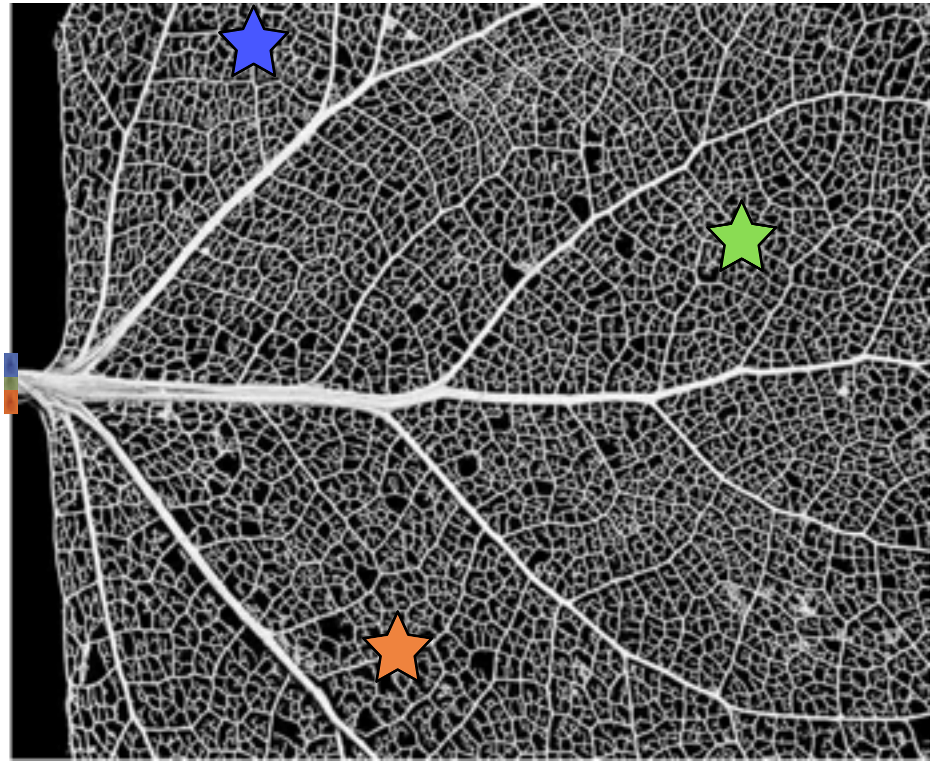
A. Becker, E.D. Demaine, S.P. Fekete, J. McLurkin:
Particle Computation: Controlling Robot Swarms with only Global Signals,
NEW To appear in: 2014 IEEE International Conference on Robotics and Automation (ICRA 2014)

combining theory and practice

Part 4.1: Why Obstacles Are a Nuisance

Obstacles as Opponents

- Targets may not be easy to reach.
- Motion planning gets quite tricky in parallel.



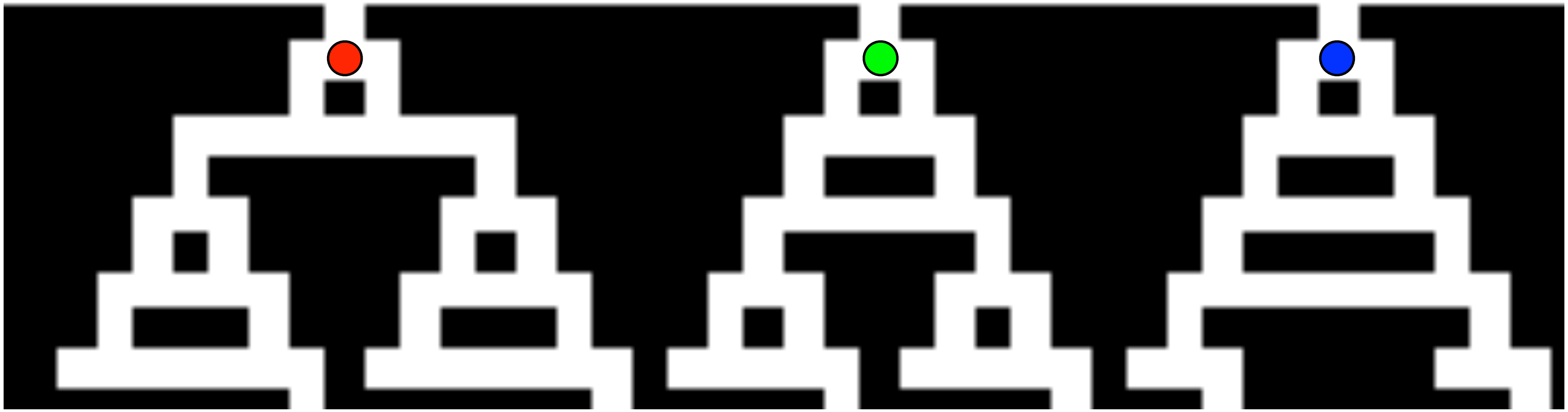
Cottonwood leaf vascular network

Complexity: Binary Variables

Choice: left or right?
Independent choices?!

Complexity: Binary Variables

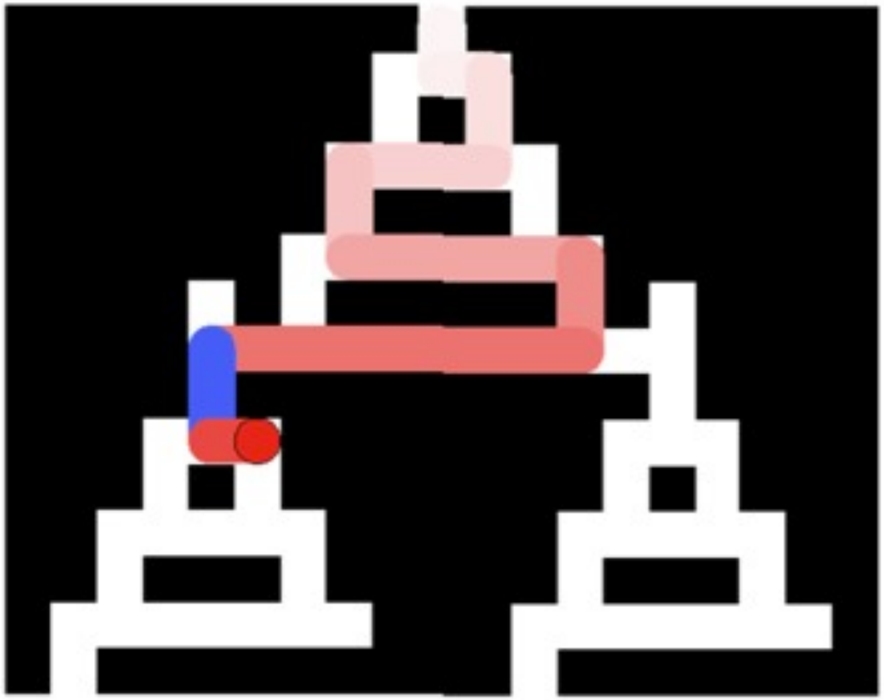
Choice: left or right?
Independent choices?!



x_2 x_3 x_4

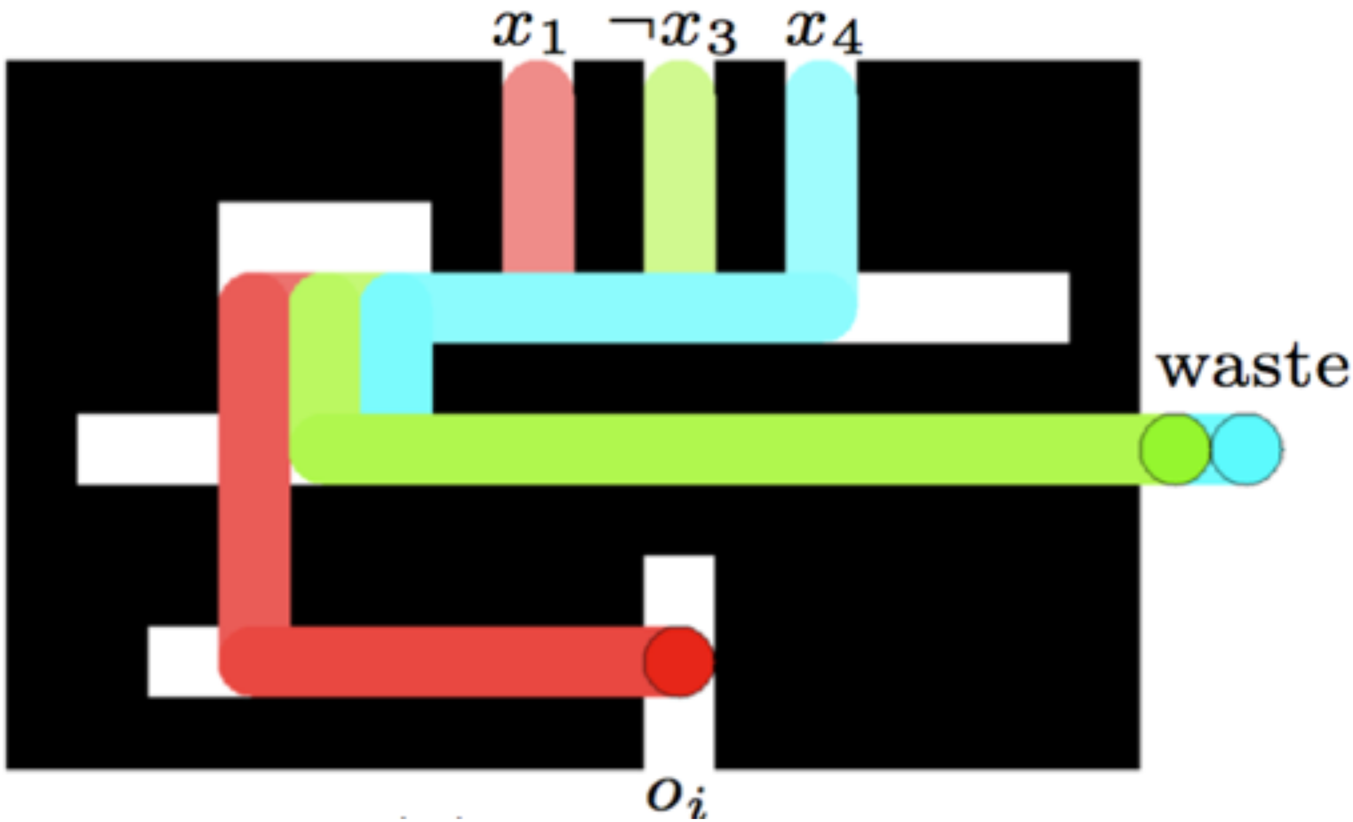
Choice only matters when it is a variable's "turn"!

Complexity: Binary Variables

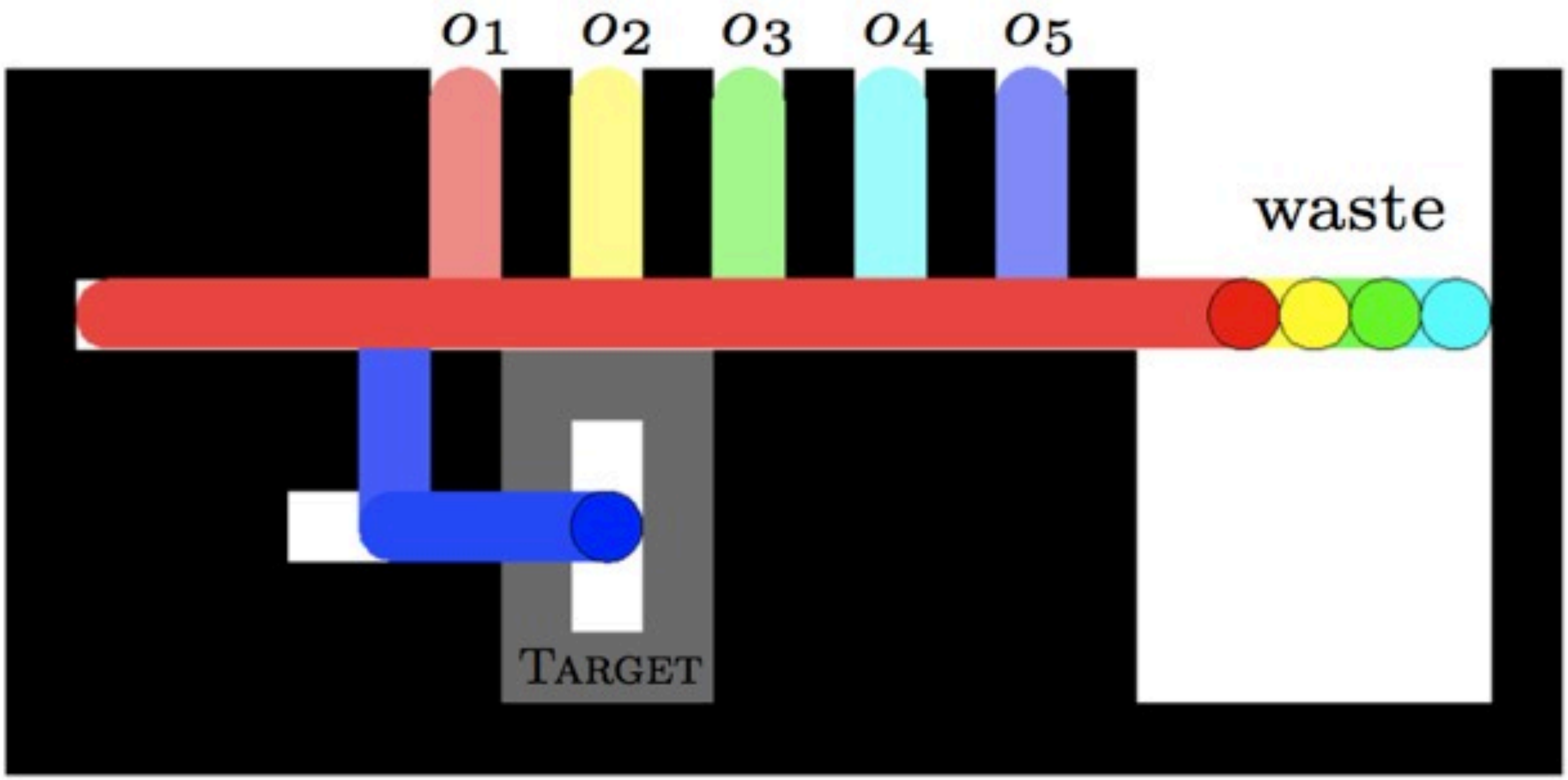


Minor detail: Avoid reversible choices!

Complexity: Clauses



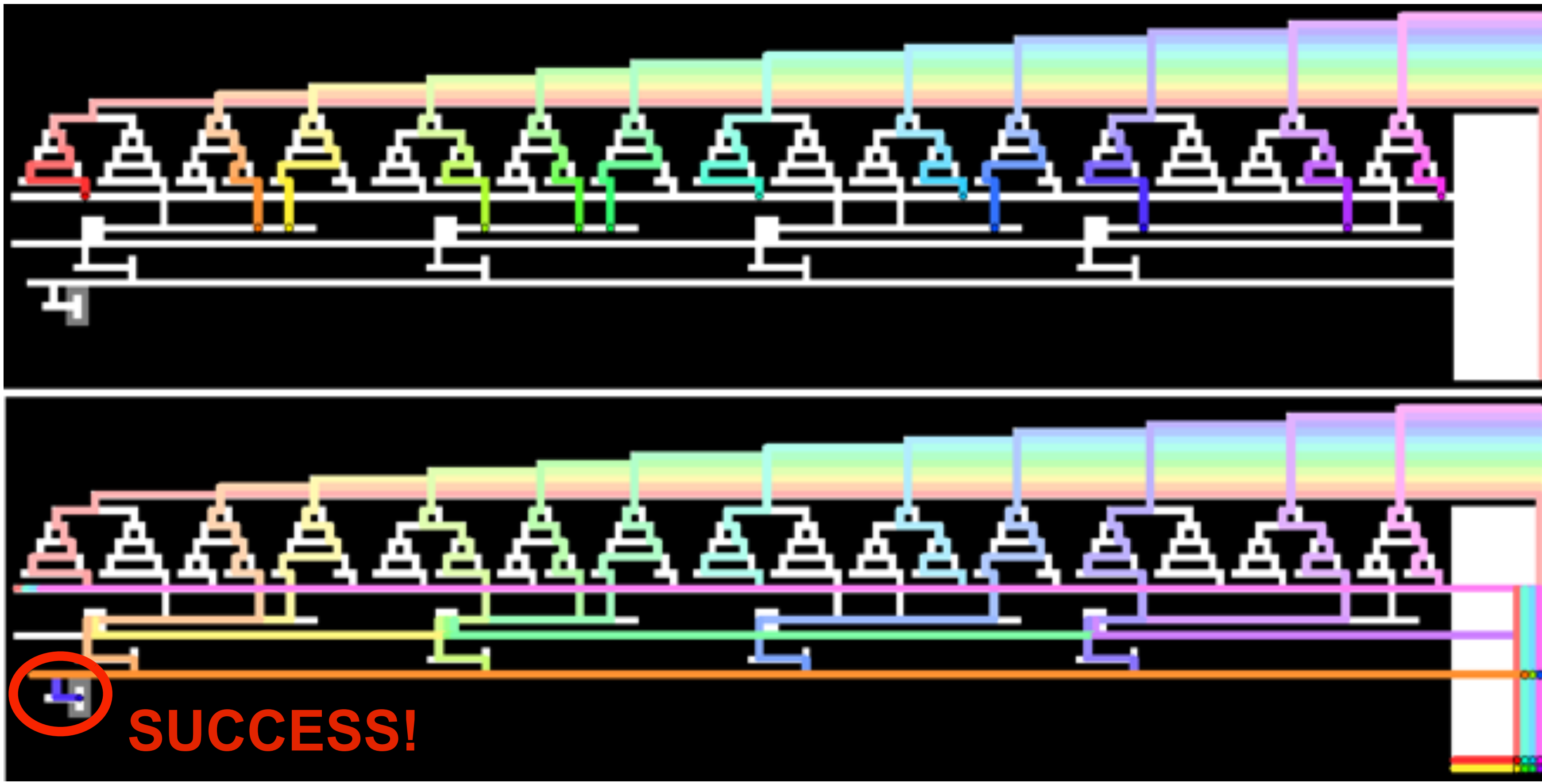
Complexity: Truth Checking



Complexity: Overall Construction

$(\neg x_1 \vee \neg x_3 \vee x_4) \wedge (\neg x_2 \vee \neg x_3 \vee x_4) \wedge (\neg x_1 \vee x_2 \vee x_4) \wedge (x_1 \vee \neg x_2 \vee x_3)$

$x_1 = 1, x_2 = 0, x_3 = 0, x_4 = 1$



SUCCESS!

Complexity: Summary

Theorem 1. GLOBALCONTROL-MANYPARTICLES *is NP-hard: given an initial configuration of movable particles and fixed obstacles, it is NP-hard to decide whether any particle can be moved to a specified location.*

Part 4.2: Why Obstacles Are a Blessing

Life without Obstacles

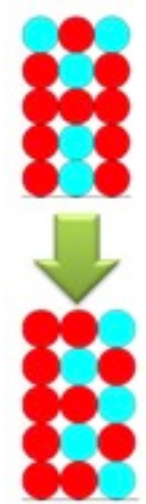
Lack of obstacles can be harmful!

Life without Obstacles

Lack of obstacles can be harmful!

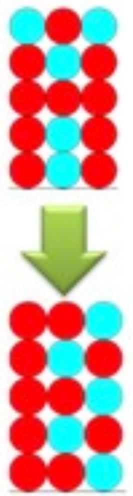
Life without Obstacles

Lack of obstacles can be harmful!



Life without Obstacles

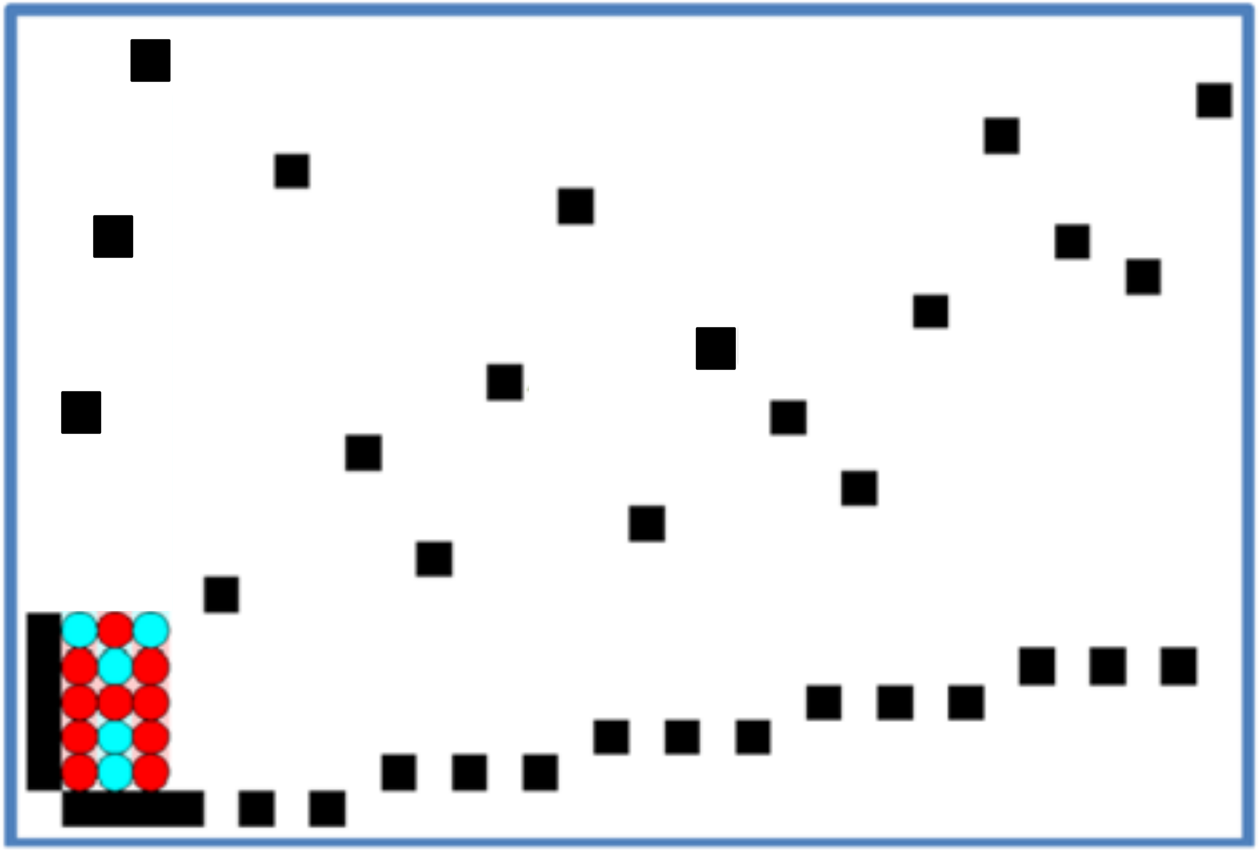
Lack of obstacles can be harmful!



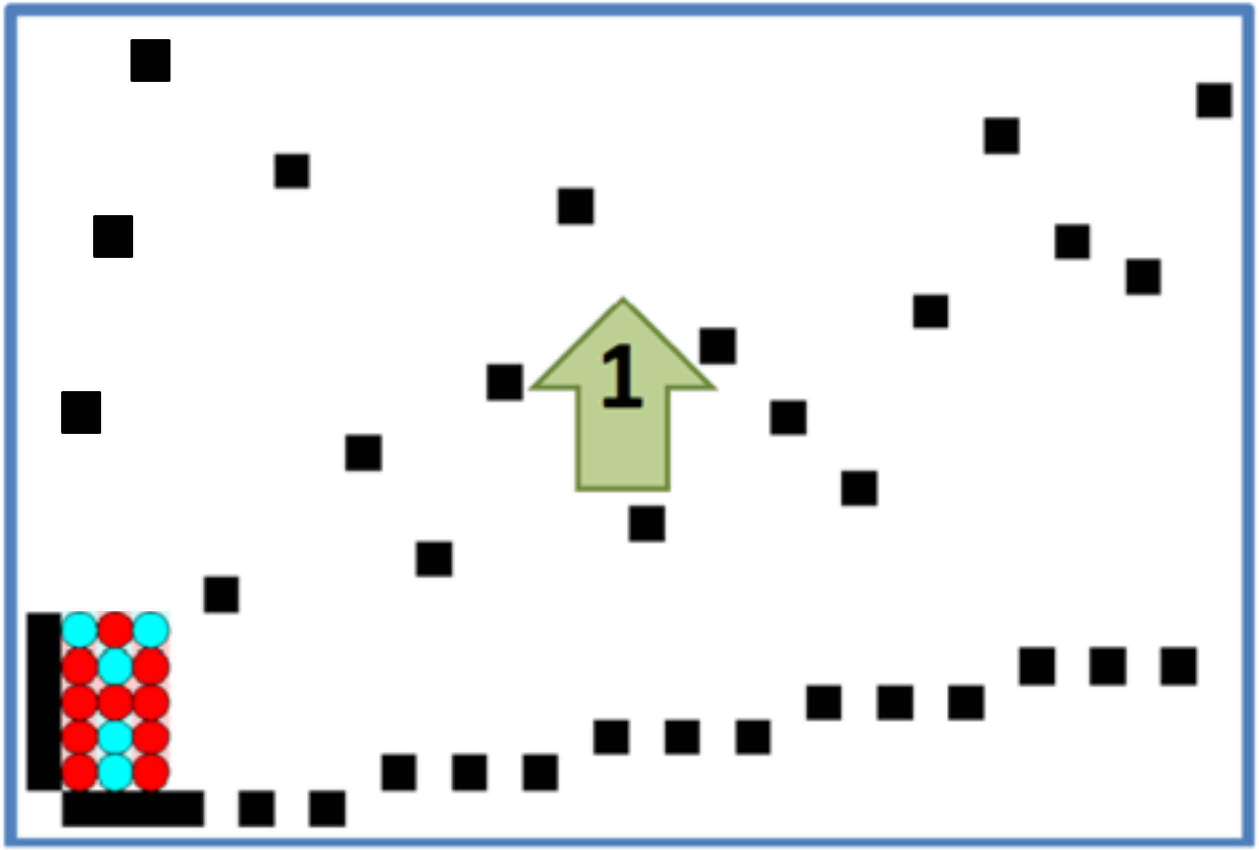
Life without Obstacles

Lack of obstacles can be harmful!

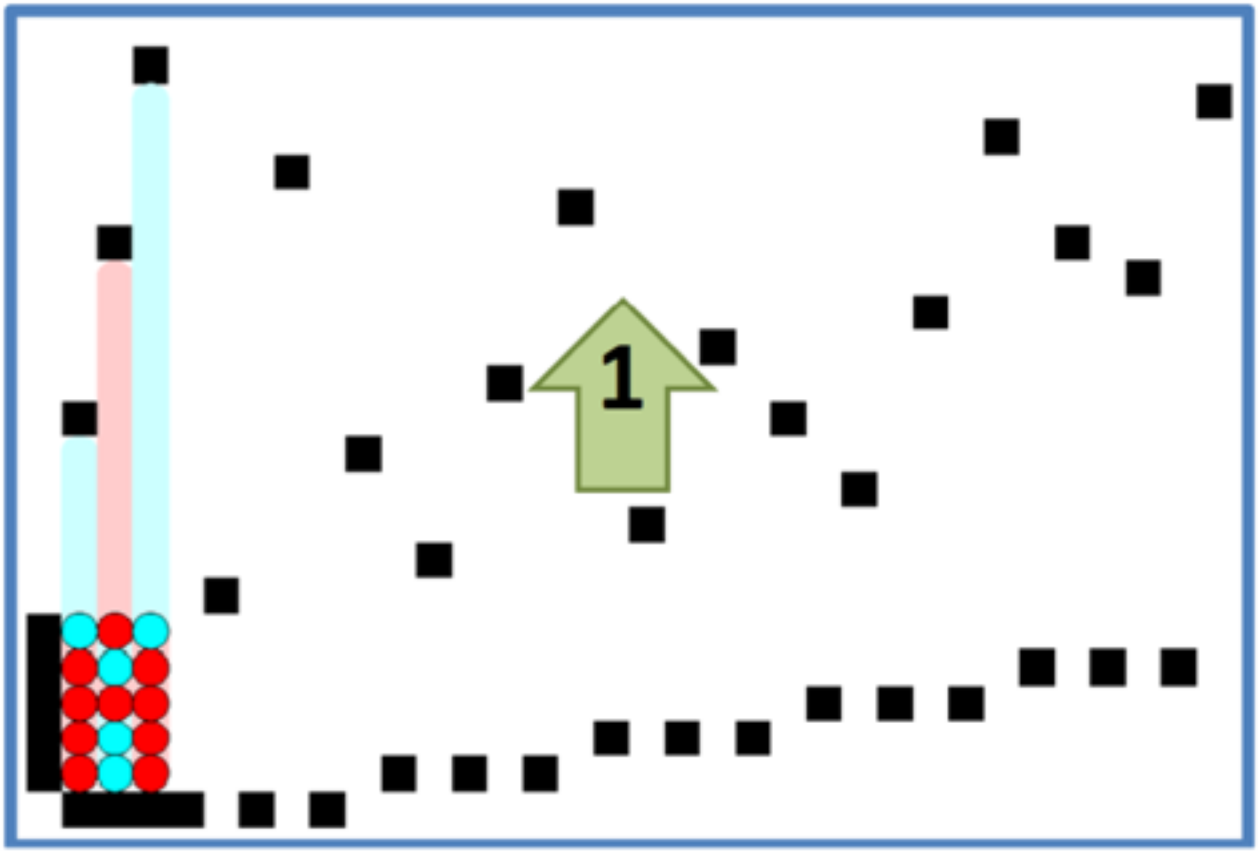
How Obstacles Can Be Helpful



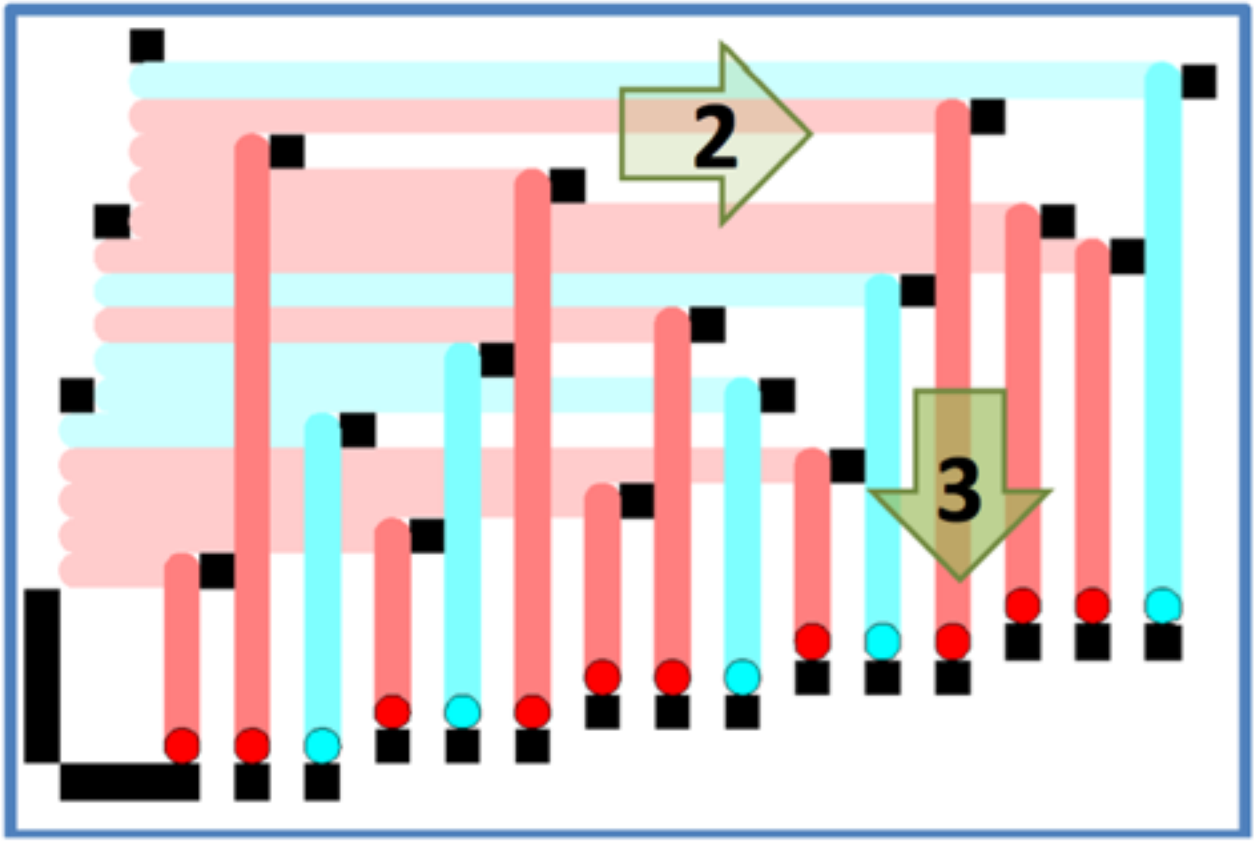
How Obstacles Can Be Helpful



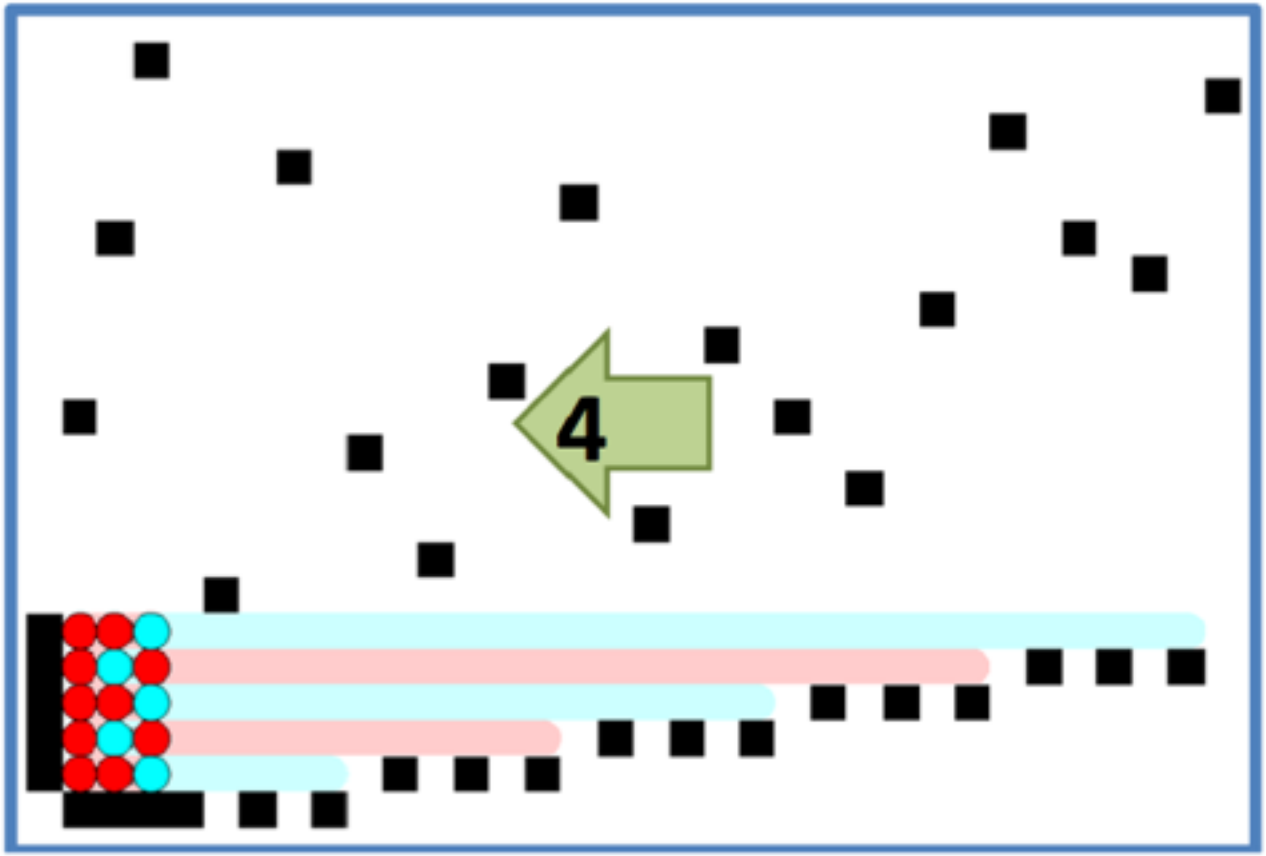
How Obstacles Can Be Helpful



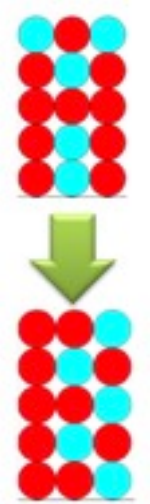
How Obstacles Can Be Helpful

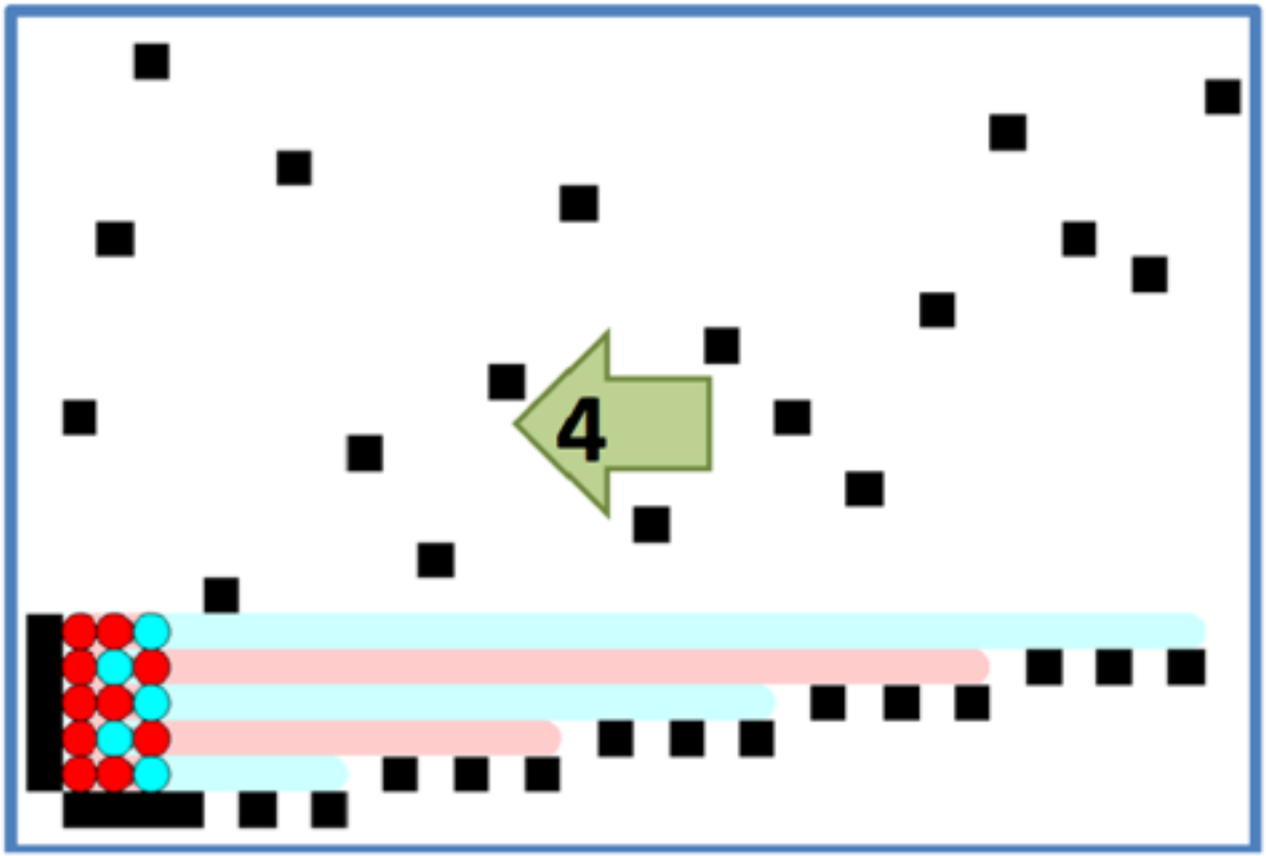


How Obstacles Can Be Helpful

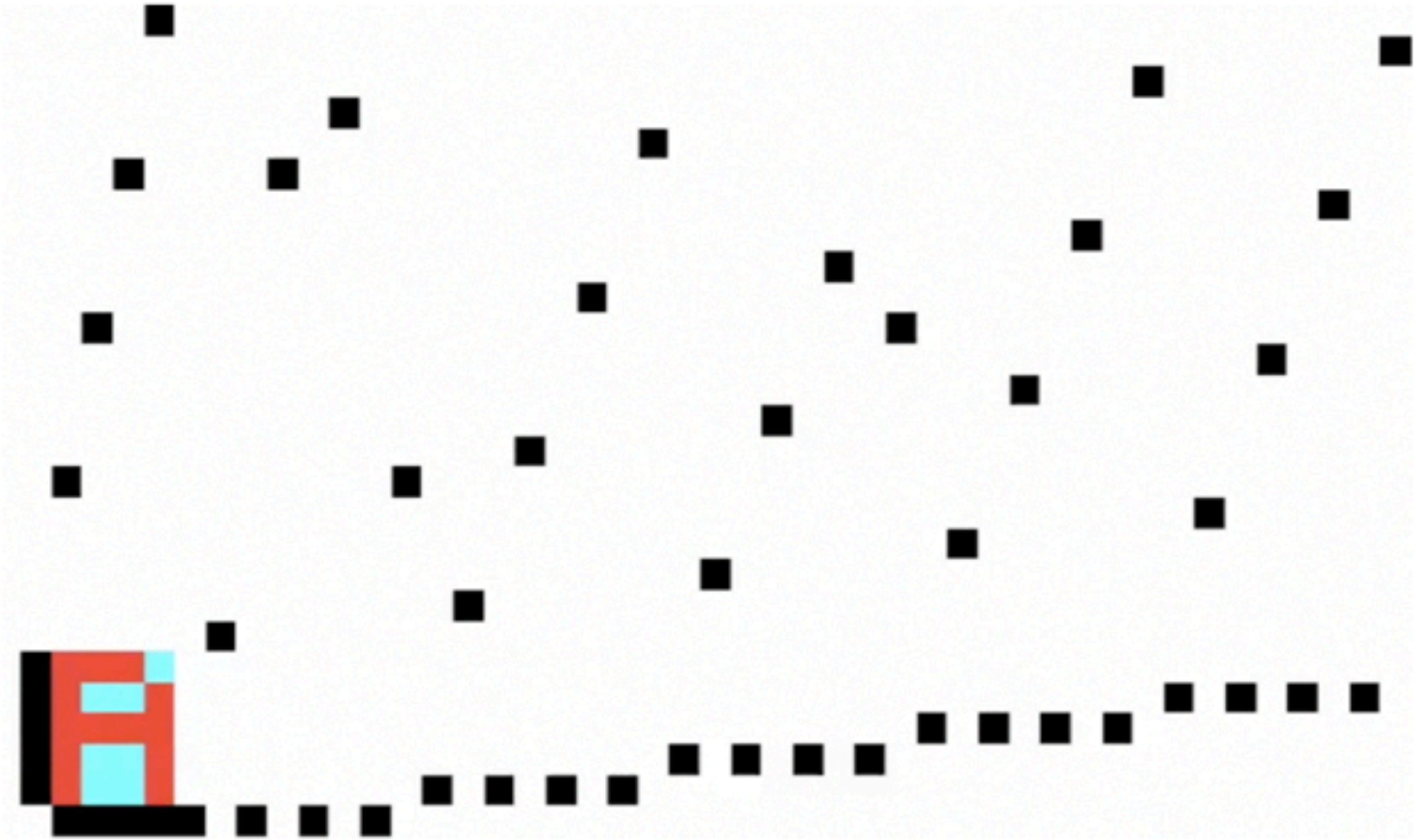


How Obstacles Can Be Helpful

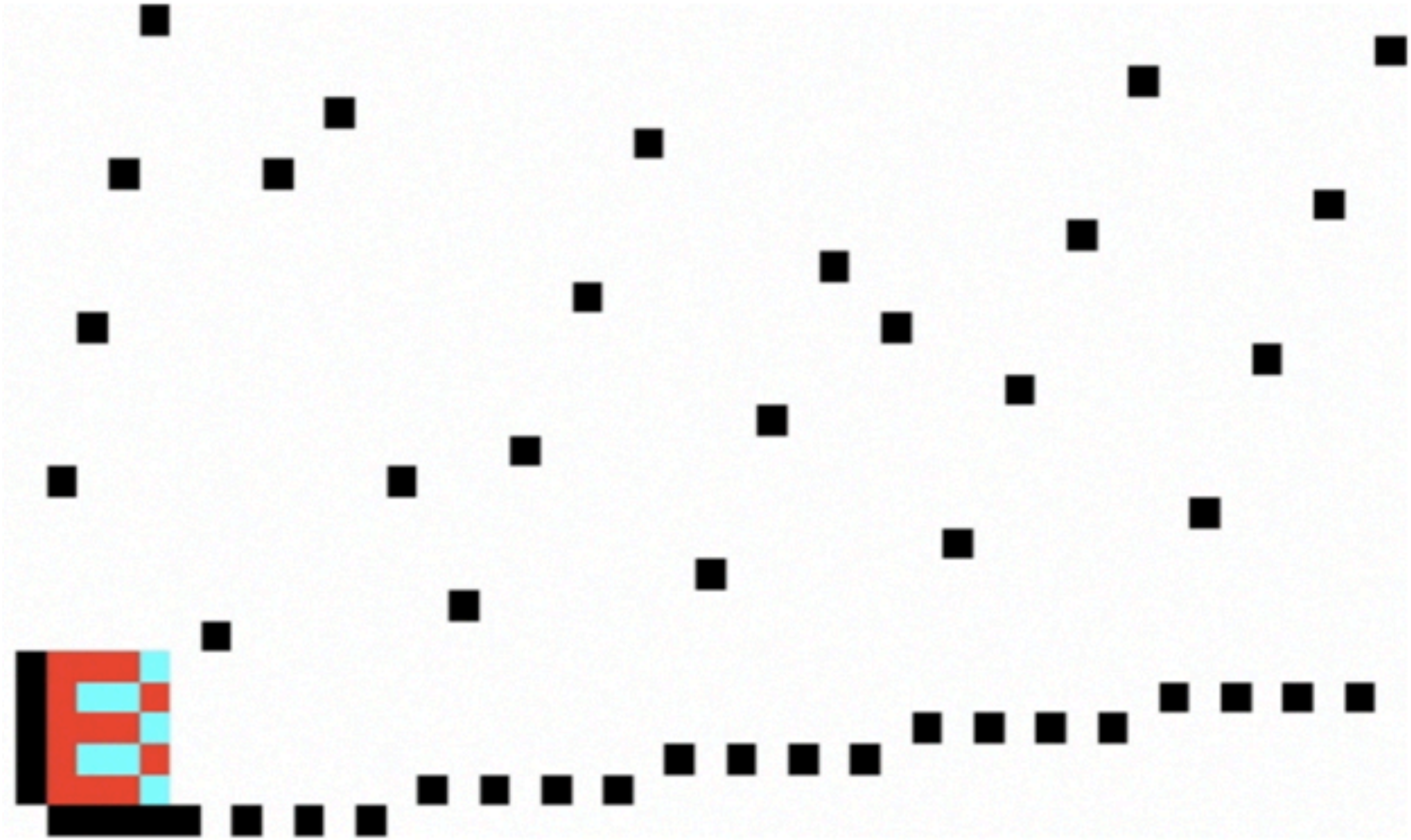




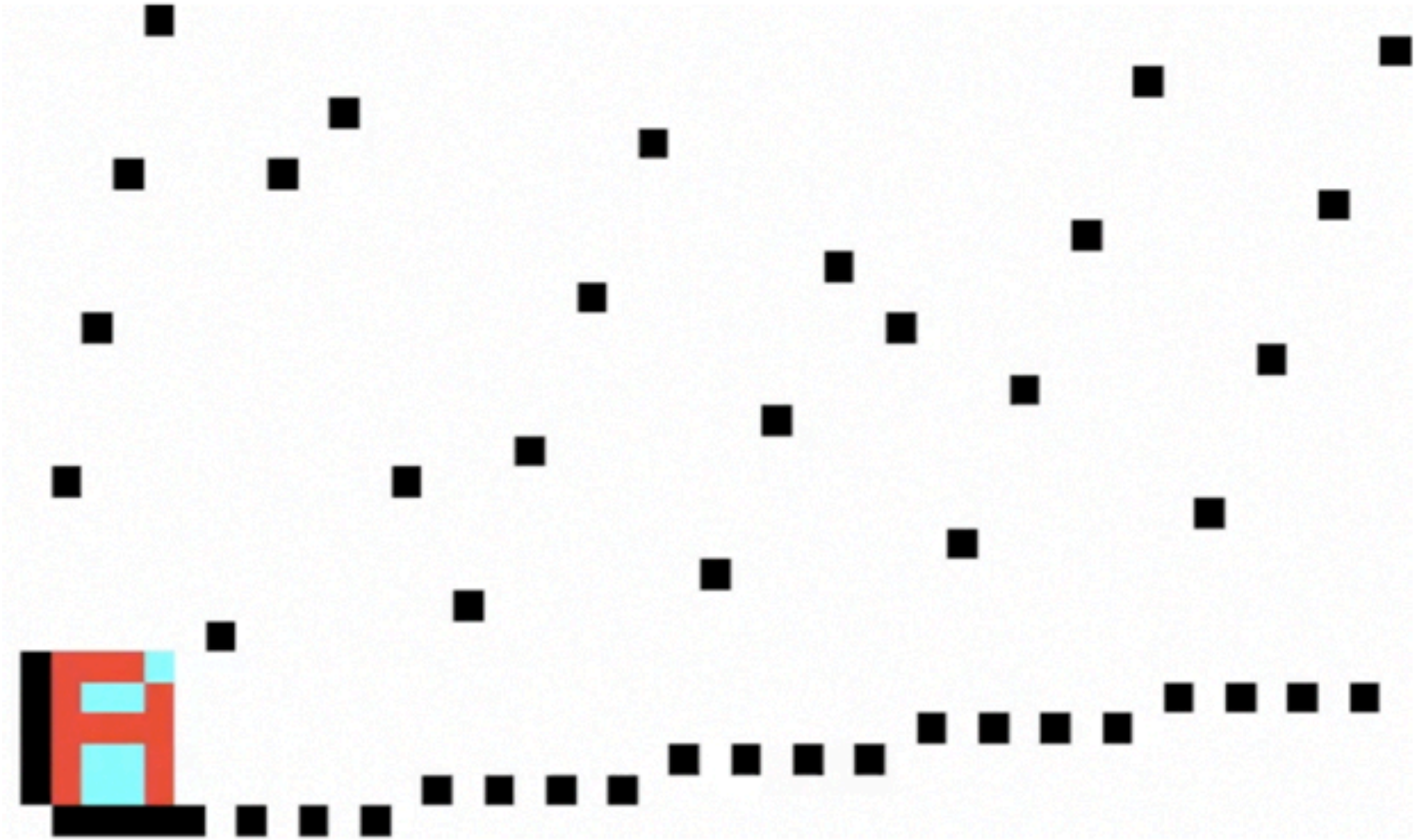
More Obstacle Action!



More Obstacle Action!

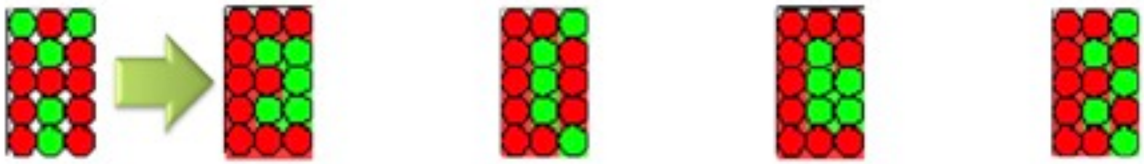


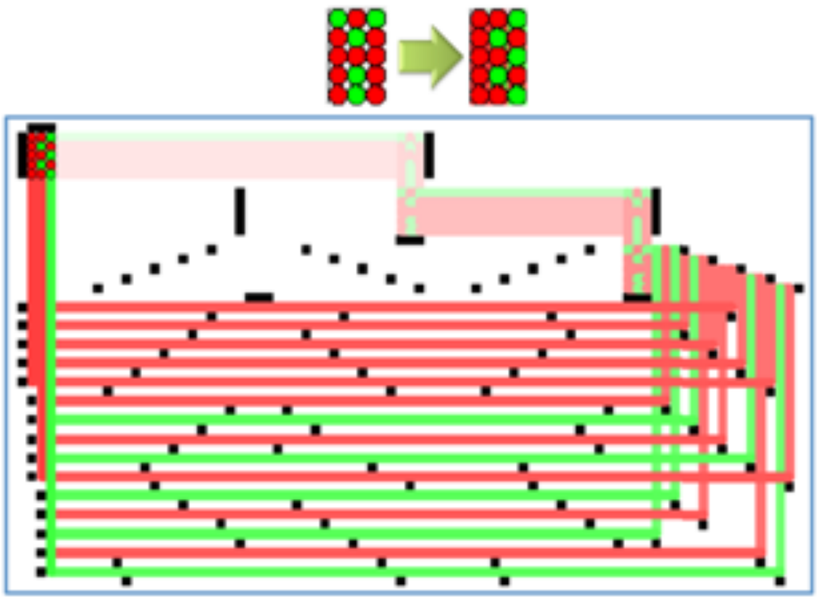
More Obstacle Action!

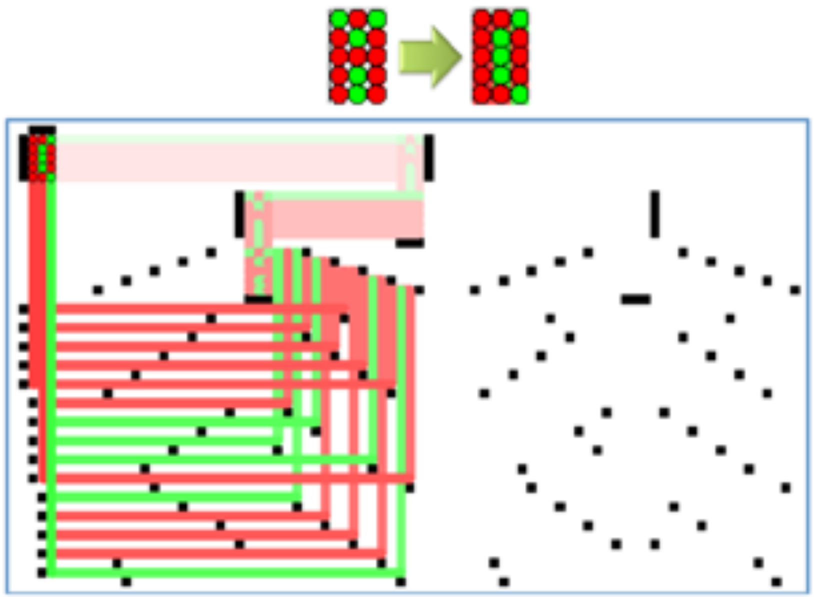


Multiple Permutations

Multiple Permutations

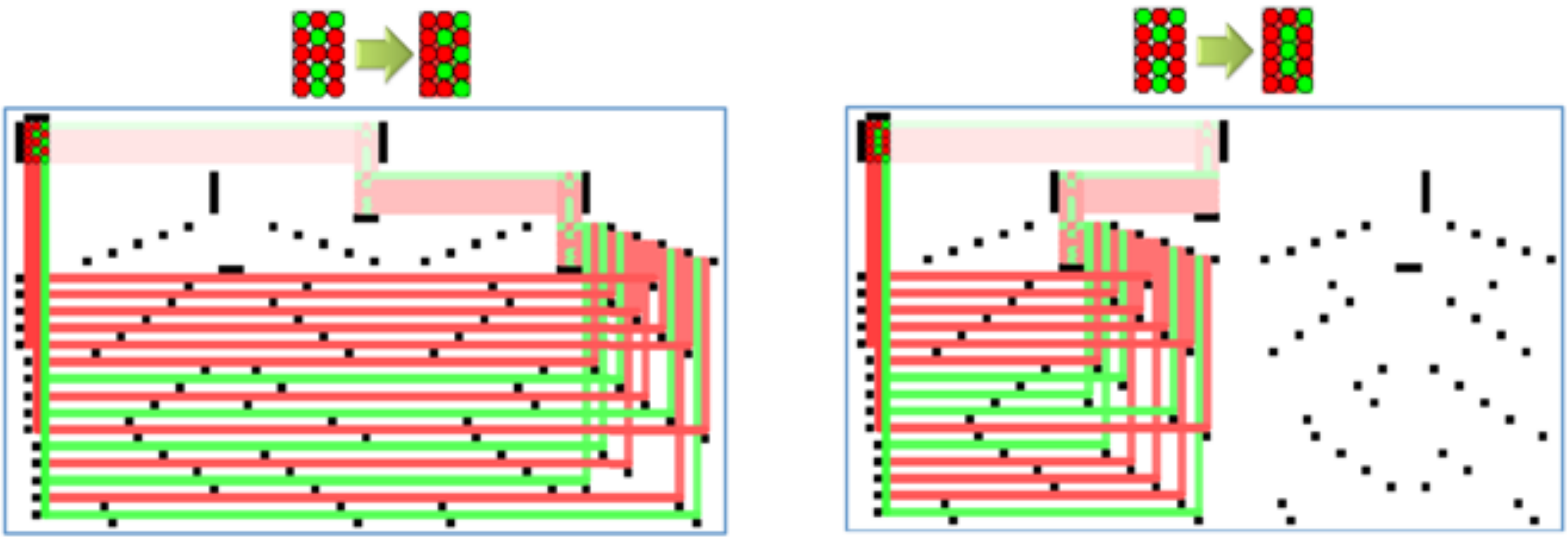




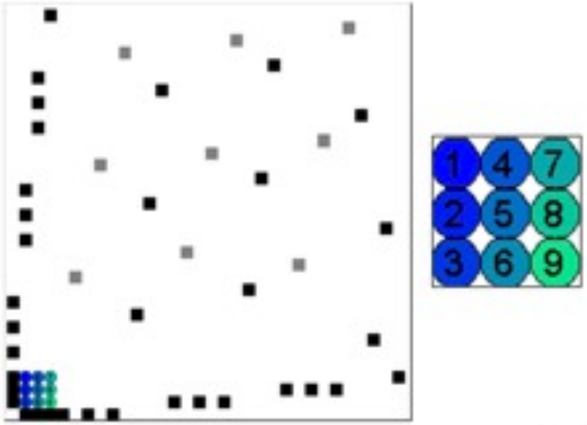


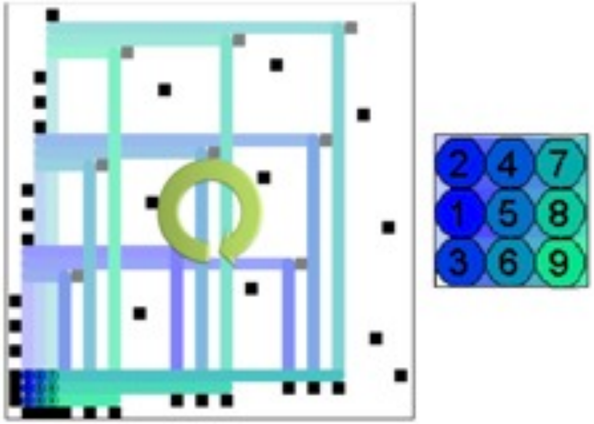
Multiple Permutations

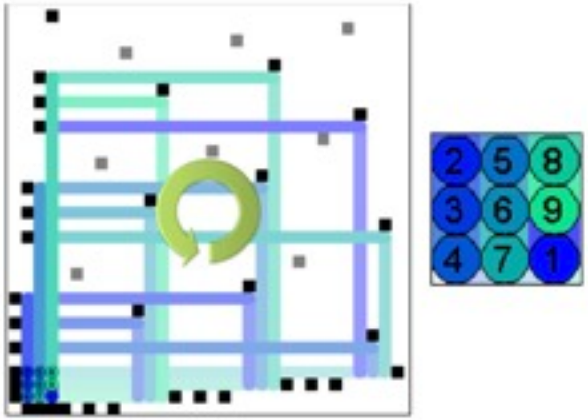
Theorem 3. *For any set of k fixed, but arbitrary, permutations of $n \times n$ pixels, we can construct a set of $O(kN)$ obstacles, such that we can switch from a start arrangement into any of the k permutations using at most $O(\log k)$ force-field moves.*



Designing Obstacles







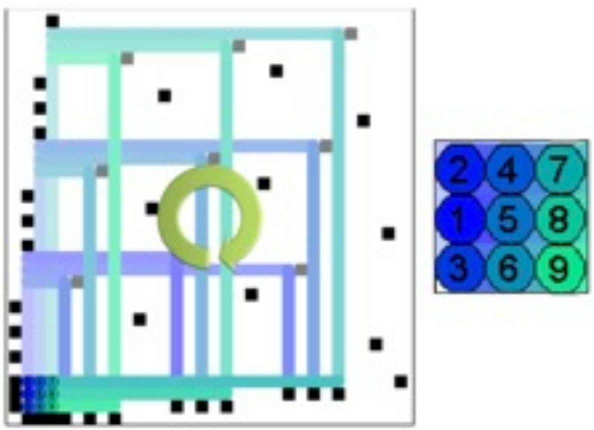
CW: (12)

CCW: (123456789)

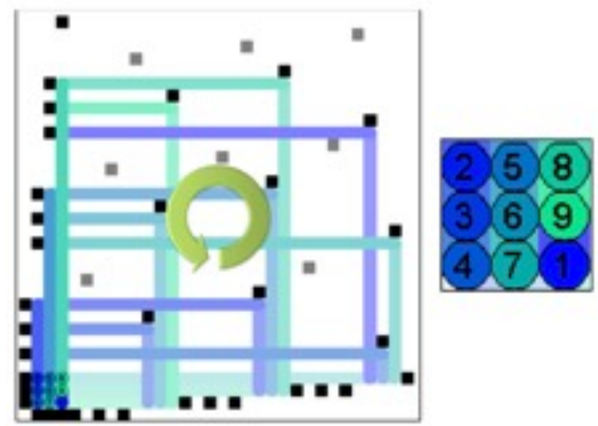
Designing Obstacles

Lemma 5. *Any permutation of N objects can be generated by the two base permutations $p = (12)$ and $q = (12 \cdots N)$. Moreover, any permutation can be generated by a sequence of length at most N^2 that consists of p and q.*

Theorem 6. *We can construct a set of $O(N)$ obstacles such that any $n \times n$ arrangement of N pixels can be rearranged into any other $n \times n$ arrangement π of the same pixels, using at most $O(N^2)$ force-field moves.*



CW: (12)



CCW: (123456789)

Designing Obstacles

Lemma 7. *Any permutation of N objects can be generated by the N base permutations $p_1 = (12), p_2 = (13), \ldots, p_{N-1} = (1(N-1))$ and $q = (12 \cdots N)$. Moreover, any permutation can be generated by a sequence of length at most N that consists of the p_i and q.*

Theorem 8. *We can construct a set of $O(N^2)$ obstacles such that any $n \times n$ arrangement of N pixels can be rearranged into any other $n \times n$ arrangement π of the same pixels, using at most $O(N \log N)$ force-field moves.*

Theorem 9. *Suppose we have a set of obstacles such that any permutation of an $n \times n$ arrangement of pixels can be achieved by at most M force-field moves. Then M is at least $\Omega(N \log N)$.*

Proof. Each permutation must be achieved by a sequence of force-field moves. Because each decision for a force-field move $\{u, d, l, r\}$ partitions the remaining set of possible permutations into at most four different subsets, we need at least $\Omega(\log(N!)) = \Omega(N \log N)$ such moves. □

More on Complexity!

Theoretical Computer Science 36 (1985) 265–289
North-Holland

THE COMPLEXITY OF FINDING MINIMUM-LENGTH GENERATOR SEQUENCES

Mark R. JERRUM

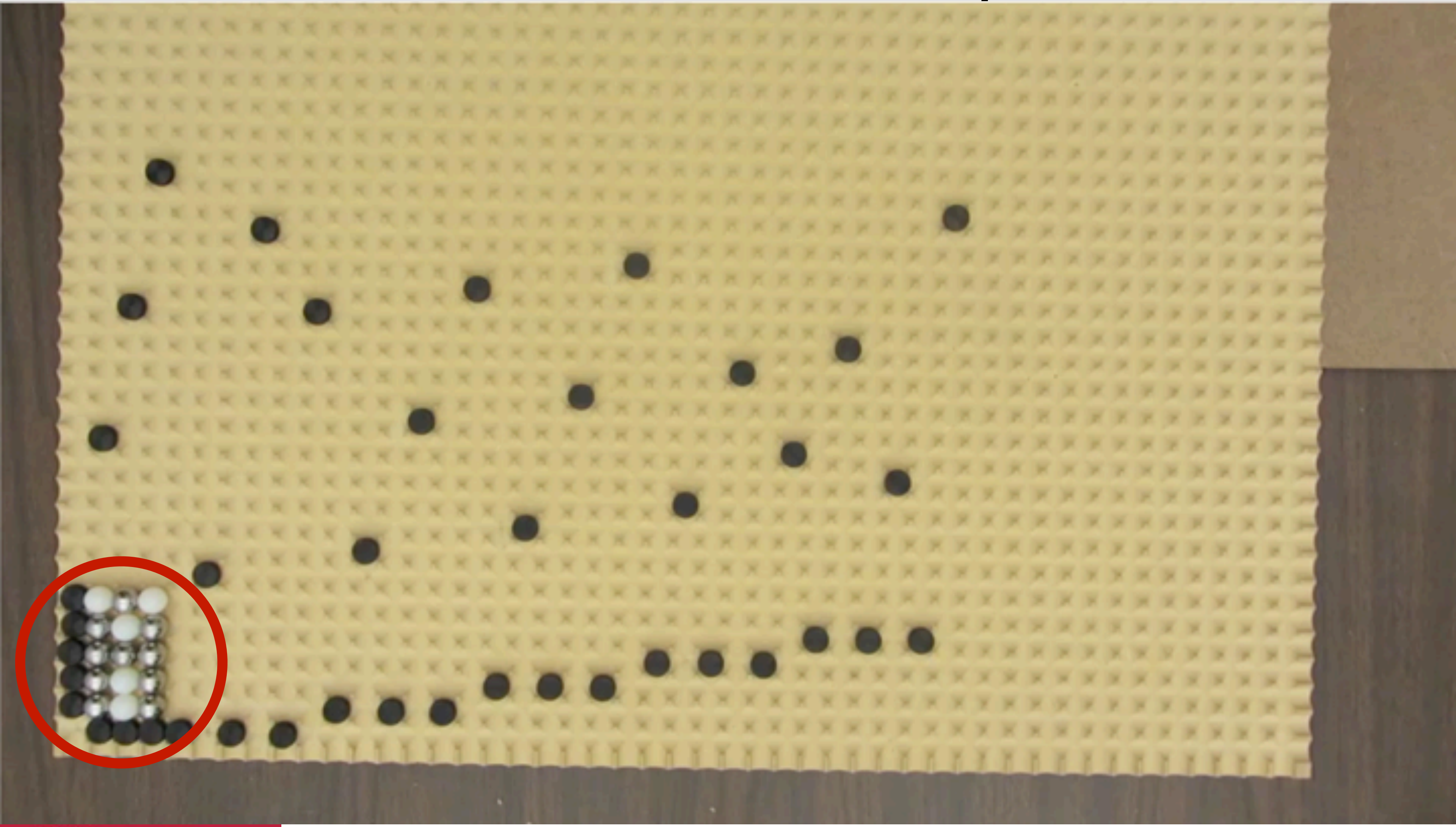
Department of Computer Science, University of Edinburgh, Edinburgh EH9 3JZ, Scotland (United Kingdom)

Communicated by M.S. Paterson
Received July 1983
Revised May 1984

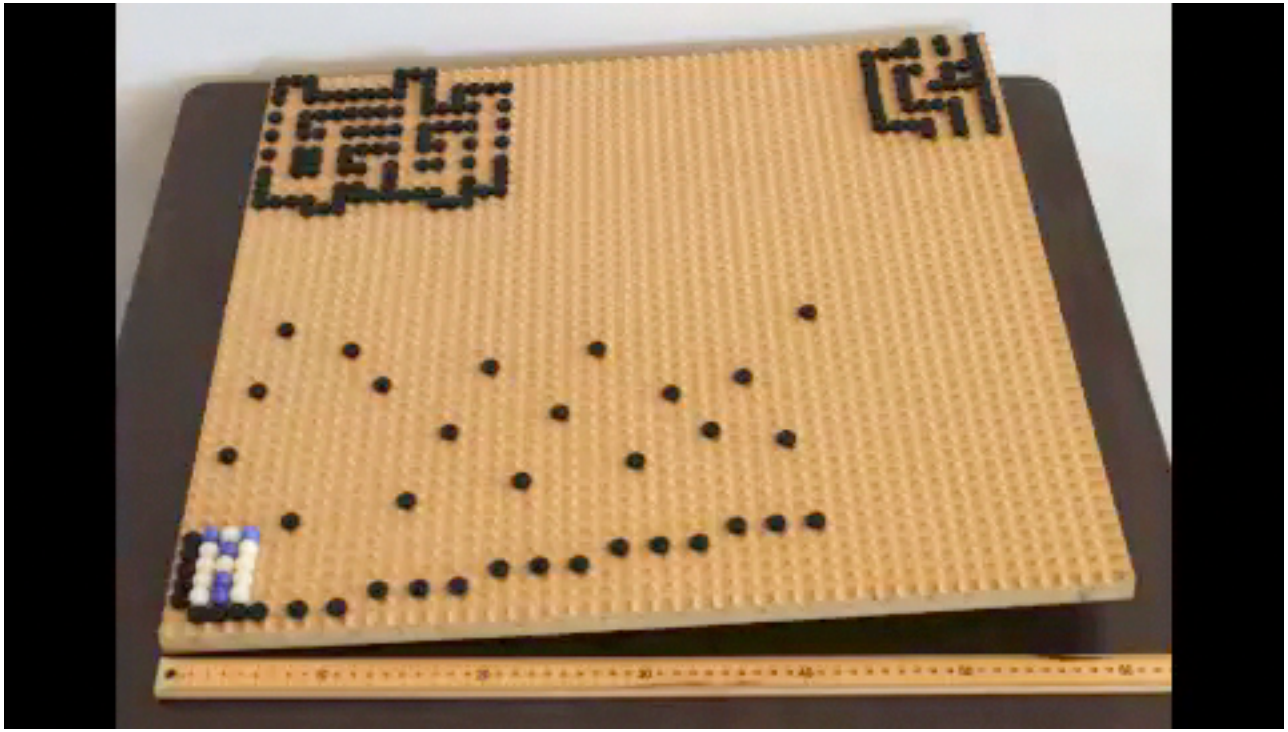
Abstract. The computational complexity of the following problem is investigated: Given a permutation group specified as a set of generators, and a single target permutation which is a member of the group, what is the shortest expression for the target permutation in terms of the generators? The general problem is demonstrated to be PSPACE-complete and, indeed, is shown to remain so even when the generator set is restricted to contain only two permutations. The restriction on generator set cardinality is the best possible, as the problem becomes soluble in polynomial time if the generator set contains only one permutation. An interesting feature of this problem is that it does not fall under the headings of 'two person games' or 'formal languages' which cover the great majority of known PSPACE-complete problems. Some restricted versions of the problem, in which the generator set is fixed rather than being part of the problem instance, are also

Part 4.3: A Real-World Demo!

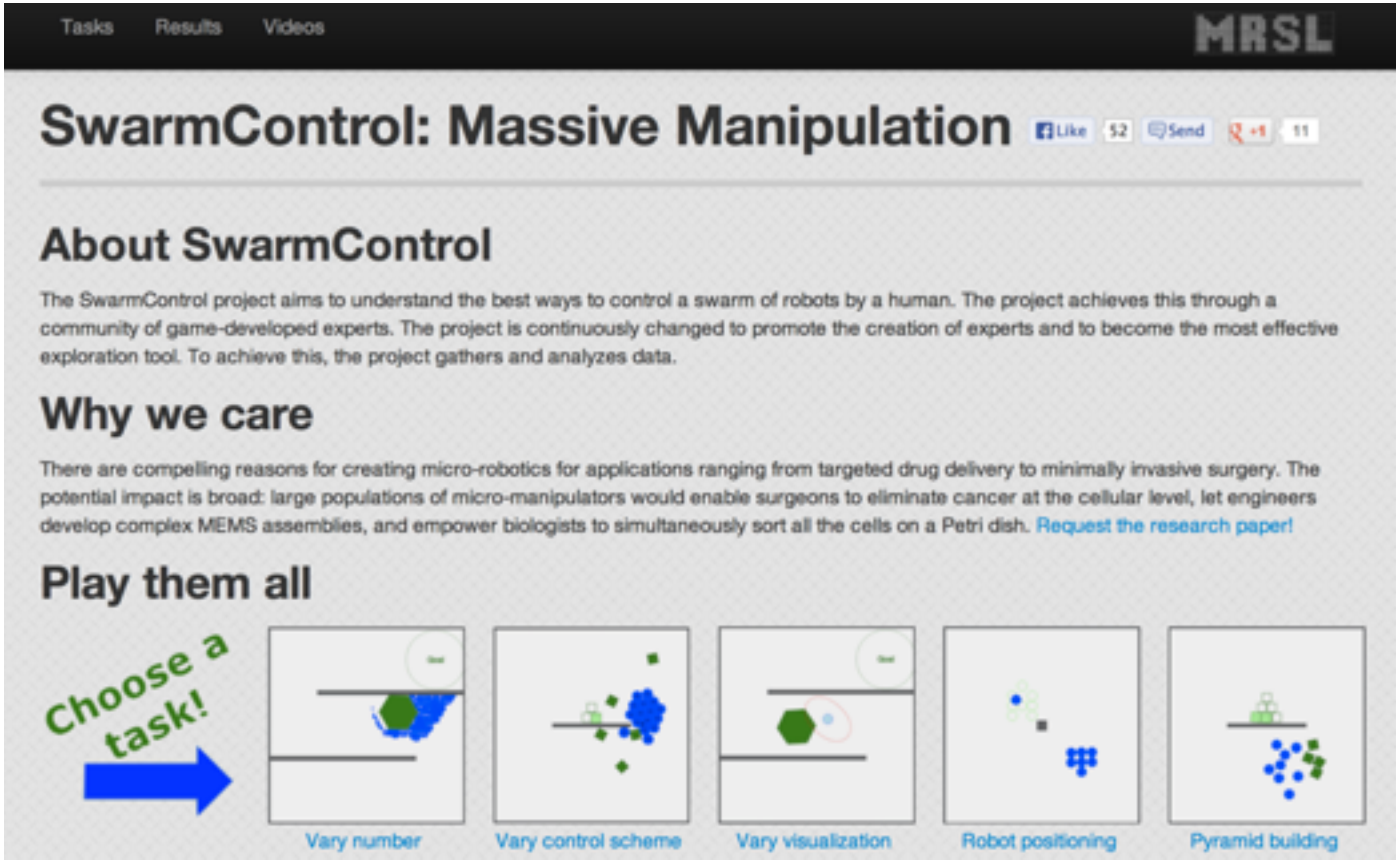
Demo with Real Objects



Demo II



Sales Pitch



http://www.swarmcontrol.net

Conclusions

- More work in theory and practice!

Thank you!