Curves and Motion

CMSC425.01 Spring 2019

Administrivia

- Final project
  - Update by next Tuesday (need to verify group membership for Elms)
  - Rubric
- Final midterm
  - Thursday Nov. 21

Final project rubric – from proposal handout

- A quality of planning and execution that can't be achieved in the last week.
- Work by all members of the team, documented by some record of your work schedule and individual contributions.
- Some innovation beyond copying an existing game, although it's not easy to be fully new in this space.
- Achievement relative to ambition. Try for something ambitious, and lack a little polish, ok. Try for less ambitious results, then make it look good.
- Non-trivial scripting, and scripts that aren't just copied as assets. Shapes and animations can be assets (although adding your own terrain or animation script would good.)

Final project rubric

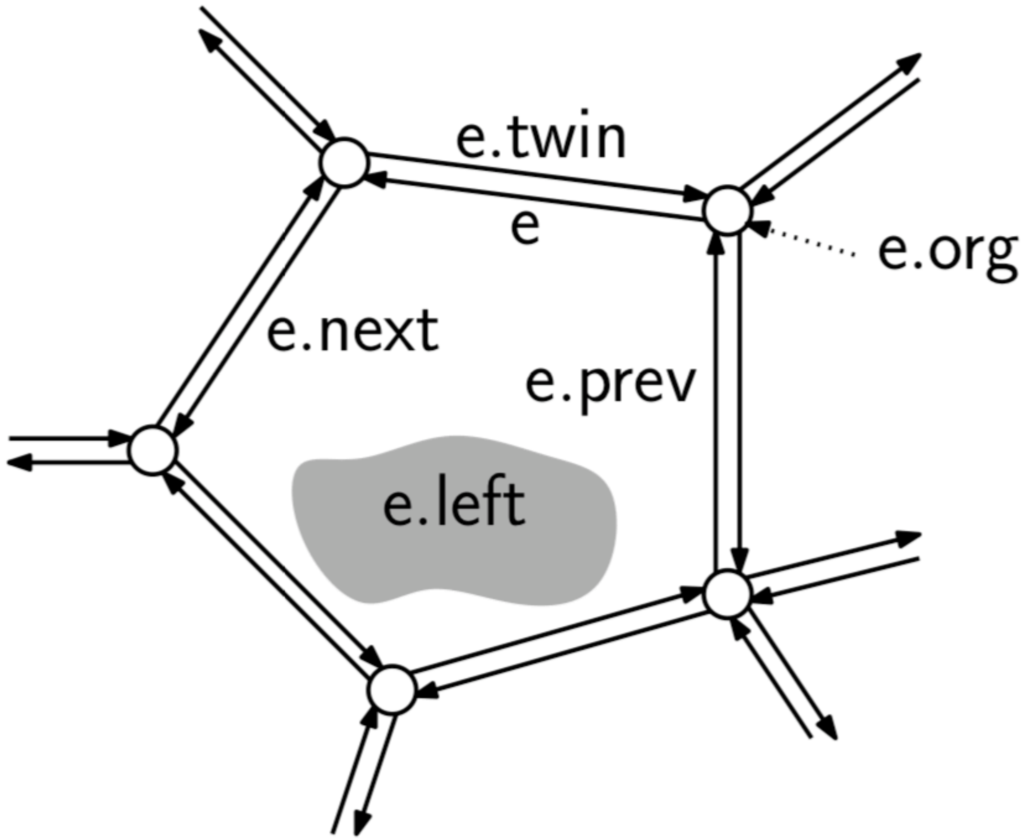
Topic	Scoring 5/5	Weights
Concept – Clear, consistent, not just copy*	/5	15
Artistic – Consistent, good look (not mixed assets)	/5	15
Algorithmic – Non-trivial scripting somewhere	/5	15
Team work – everyone contributed, documented	/5	15
Completeness – All of it works, relative to ambition*	/5	20
Group size – more people, higher expectations	/5	10
Video – video is submitted, clear	/5	5
Report – report is submitted, clear and complete	/5	5
Intangibles – instructor overall opinion	/5	5
Total		100

Asterisk *

- In your report you can spell out that
  - We did something ambitious and it didn’t quite work
  - We intentionally copied this game and here’s what we did a bit new
  - Anything else the grader should take into consideration

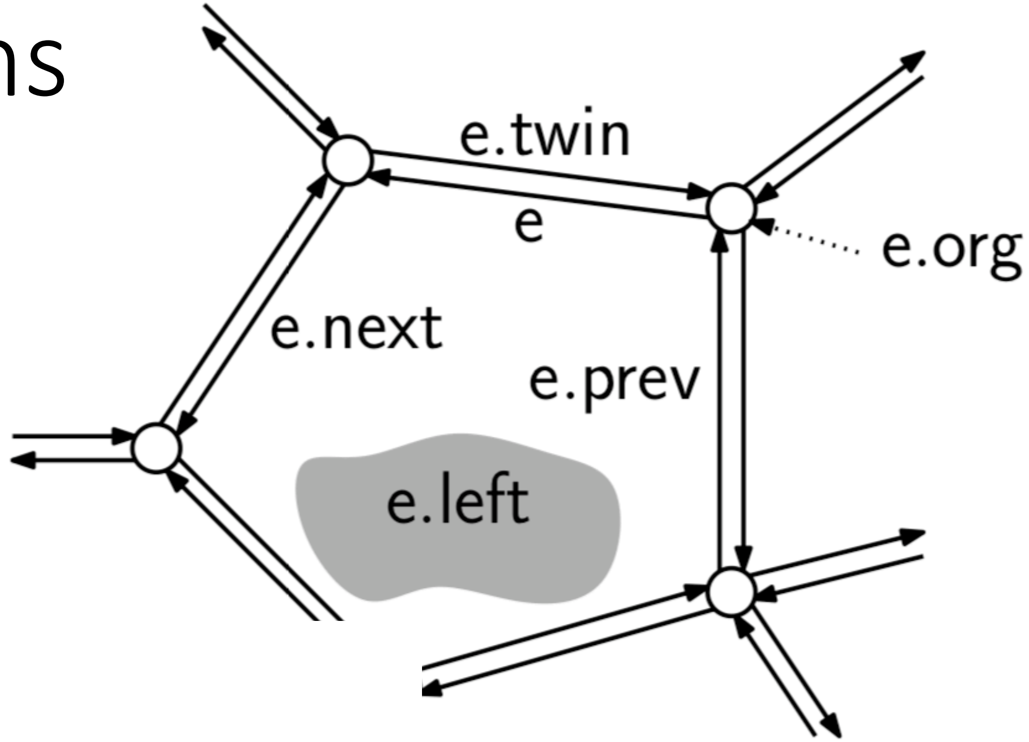
Winged edge representations

- Vertex v has coordinates plus one link to incident edge
- Face f has link to one half edge
- Edge (origin u, destination v) has
- *e.org*: e's origin
- *e.twin*: e's opposite twin half-edge
- *e.left*: the face on e's left side
- *e.next*: the next half-edge after e in counterclockwise order about e's left face
- *e.prev*: the previous half-edge to e in counterclockwise order about e's left face (that is, the next edge in clockwise order).



Winged edge representations

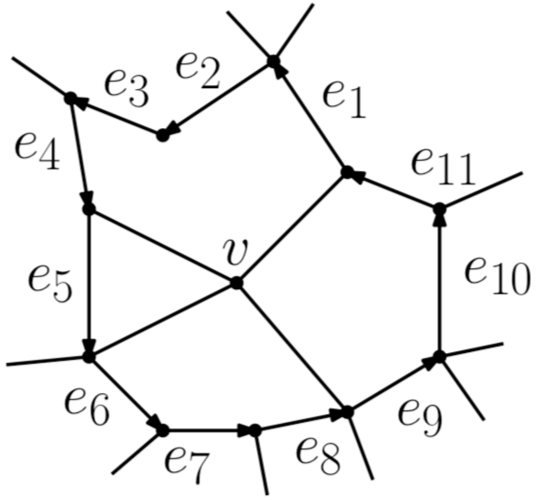
- Question: how traverse all vertices that are neighbors of v in cw order?




```
vertexNeighborsCW(Vertex v) {
    Edge start = v.incident;
    Edge e = start;
    do {
        output e.dest; // formally: output e.twin.org
        e = e.oprev; // formally: e = e.twin.next
    } while (e != start);
}
```

In class exercise

Given vertex v in a cell complex of a 2-manifold, the *link* of v is defined to be the edges that bound the faces that are incident to v, excluding the edges that are incident to v itself. Present a procedure (in pseudocode) that, given a vertex v of a DCEL, returns a list L consisting of the half edges of v's link ordered counterclockwise about v. For example, in the figure below, a possible output would be $\langle e_1, \ldots, e_{11} \rangle$. (Any cyclic permutation would be correct.)



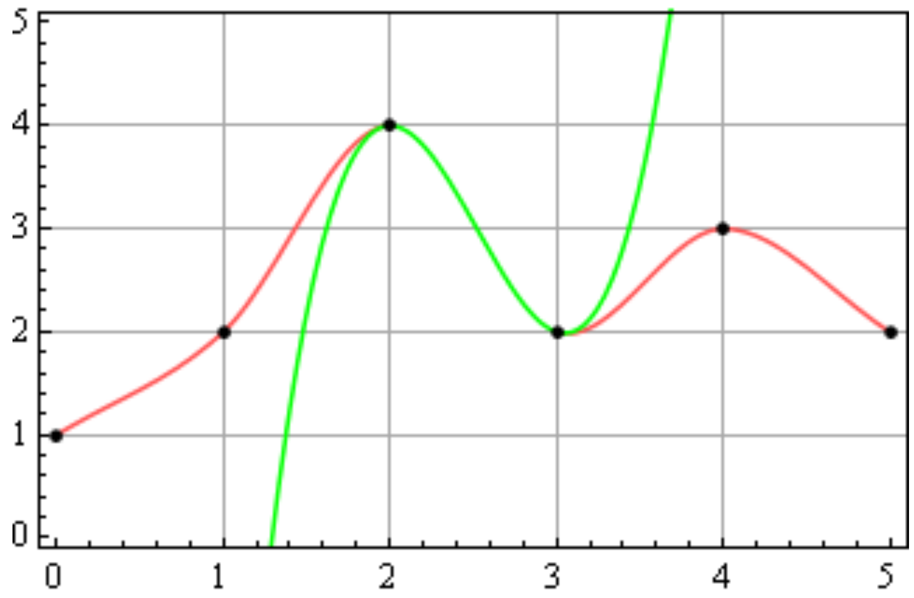
Today’s question

Curves for shapes and motion

Cubic interpolation

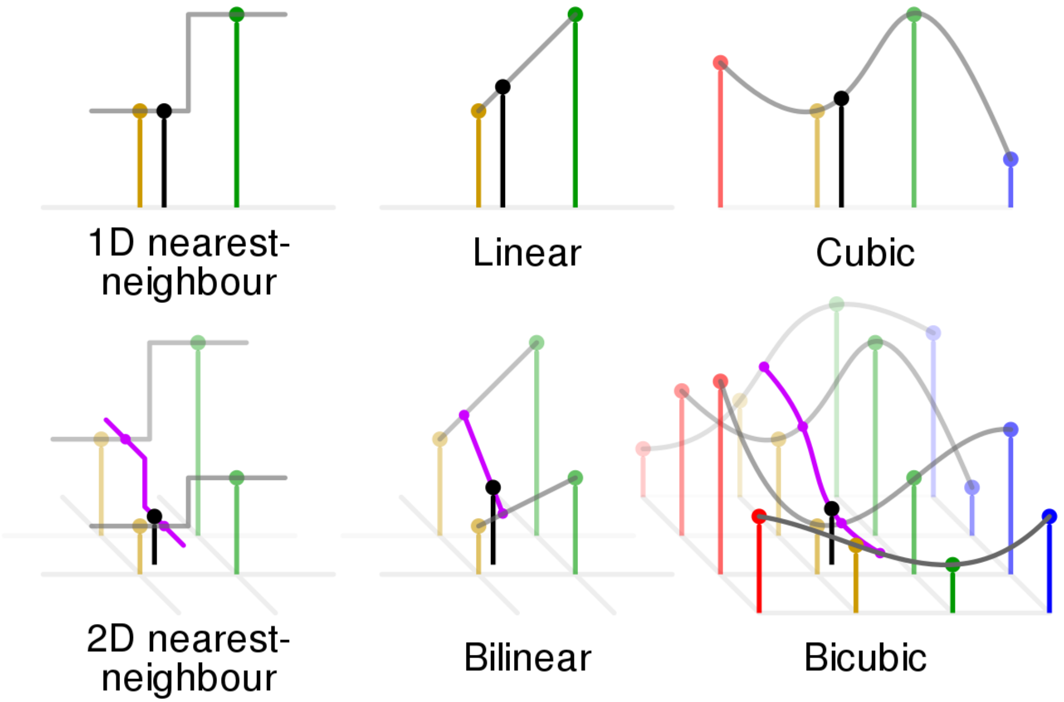
- $P(t) = ax^3 + bx^2 + cx + d$

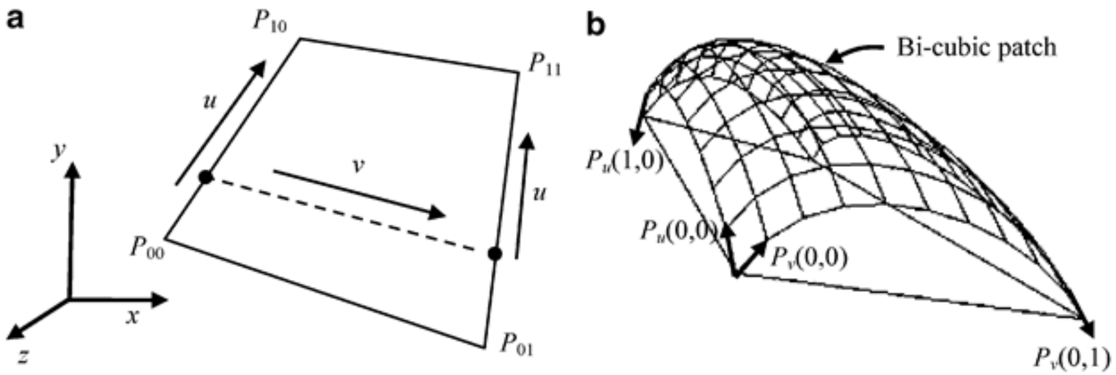
- Can match tangents at ends
- Good enough for human eye

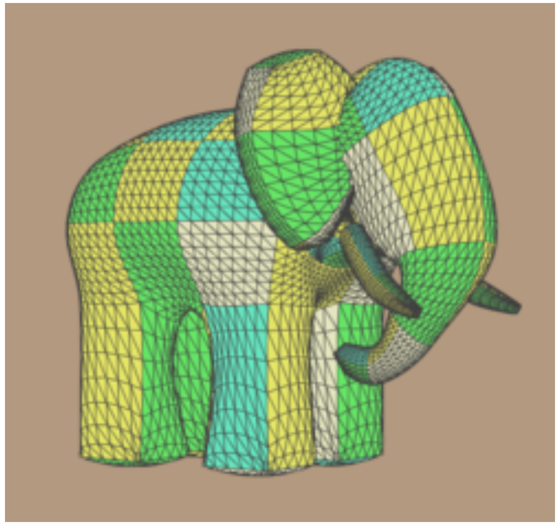


Bicubic surface patch

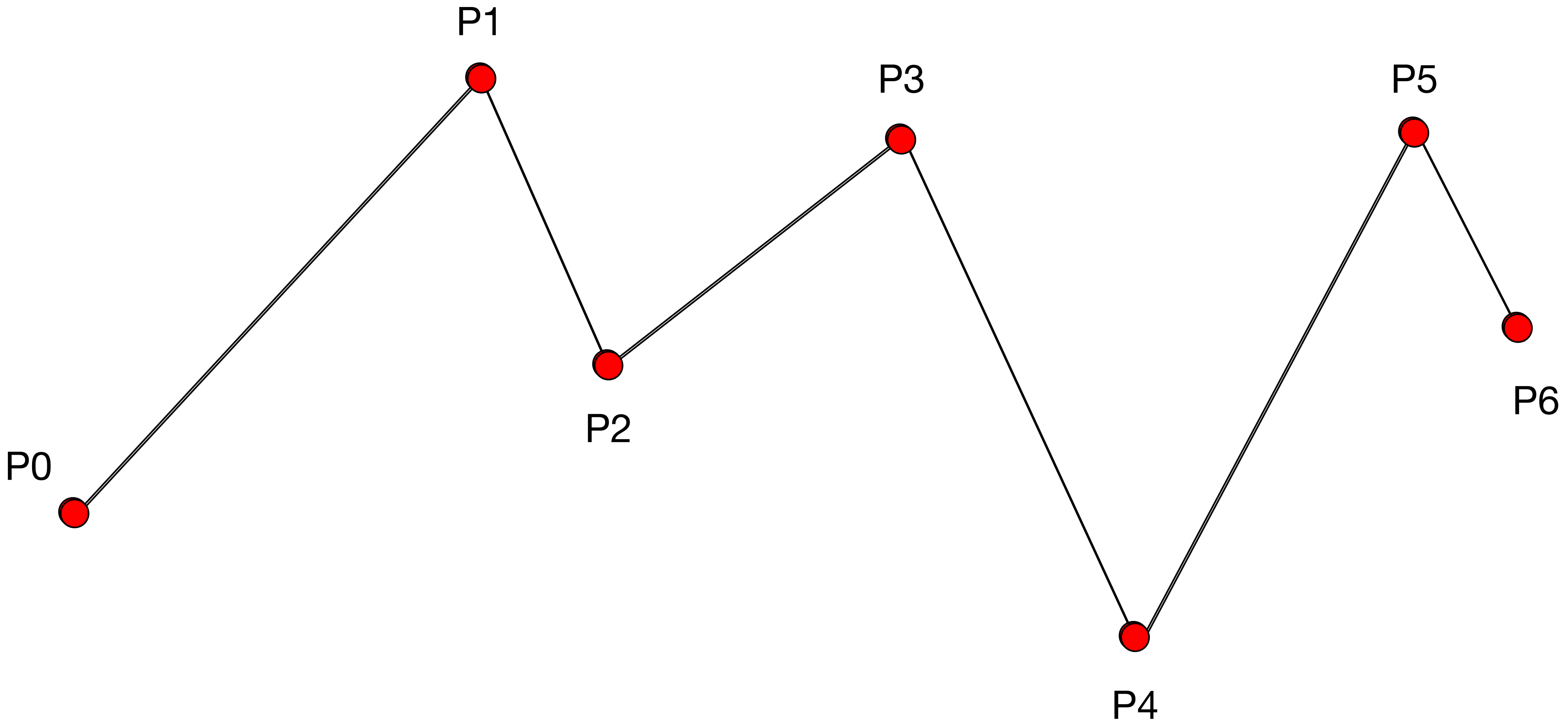
- Cubic curve in both directions





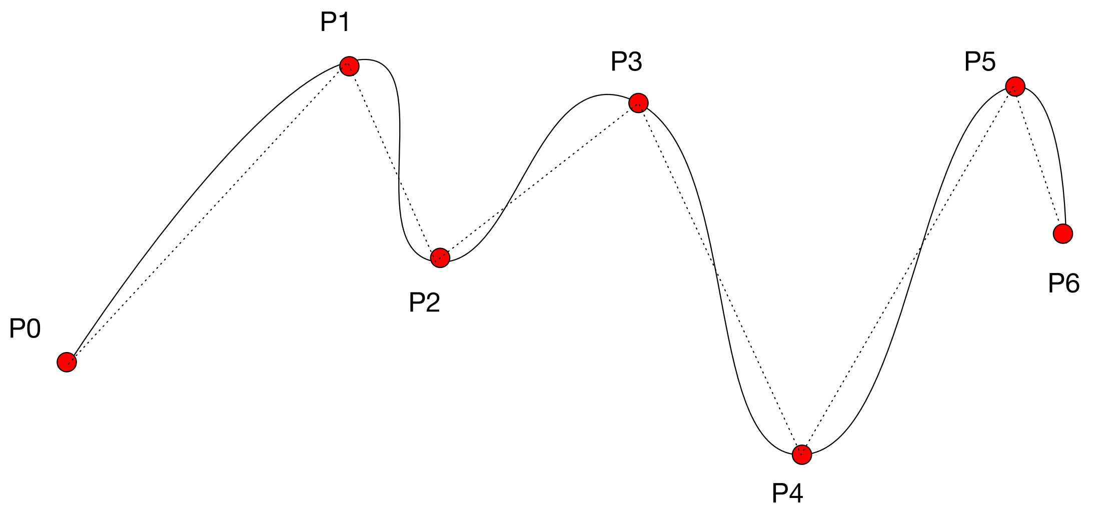


Polyline of control points

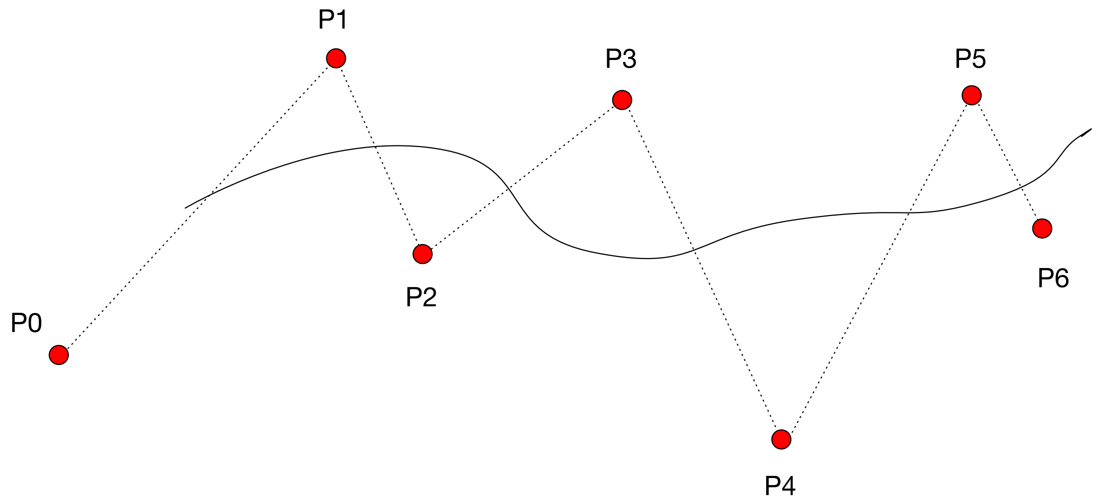


Piecewise interpolation vs. approximation

Interpolating – through points

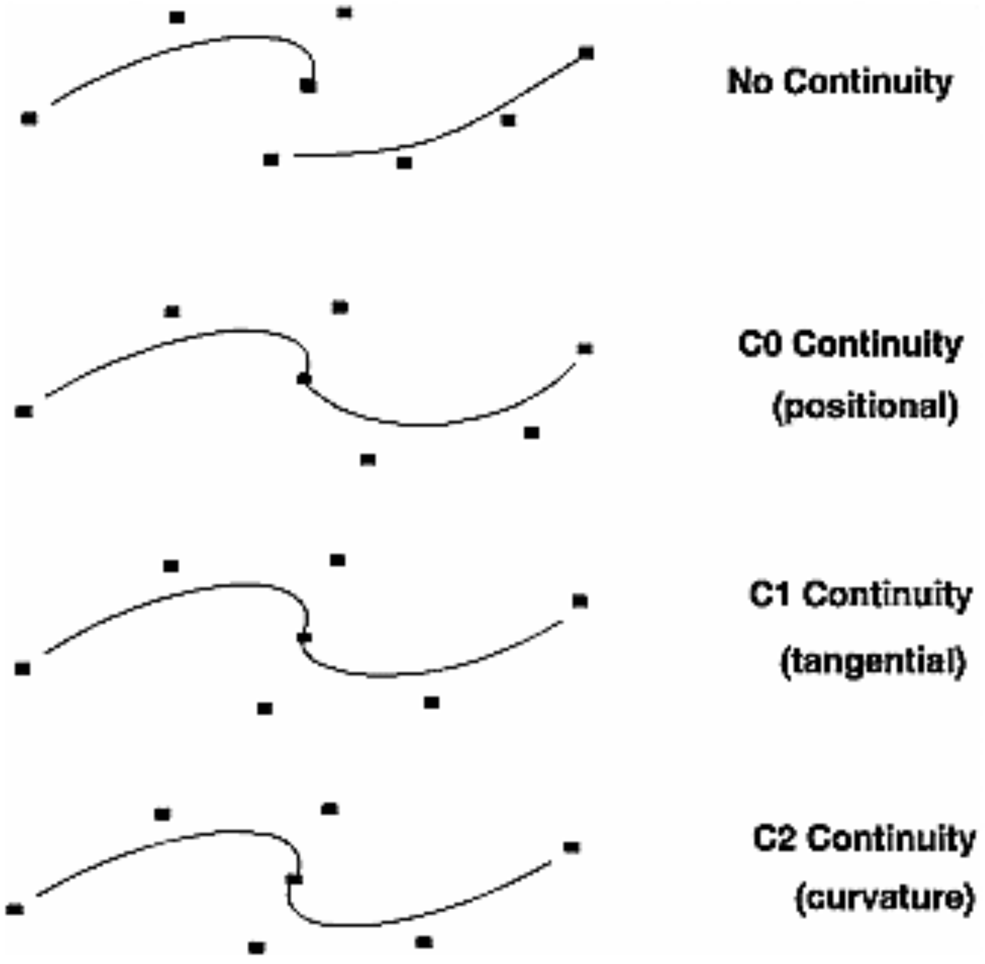


Approximating – controlled by points



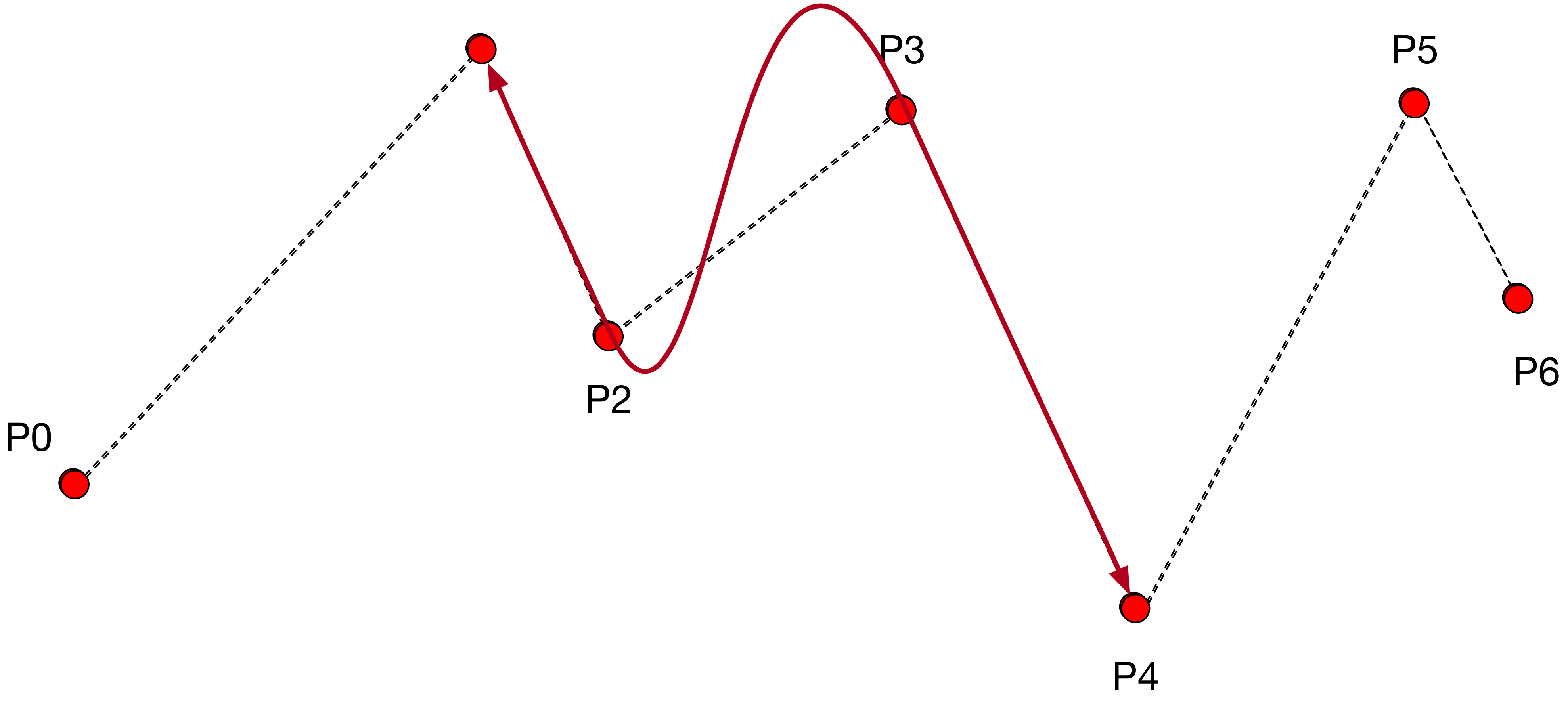
Piecewise continuity

- Continuity gives smoothness
- Applies to shape and motion
- Eg, Navmesh path
- We care about C1 continuity
  - Cubic curve enables



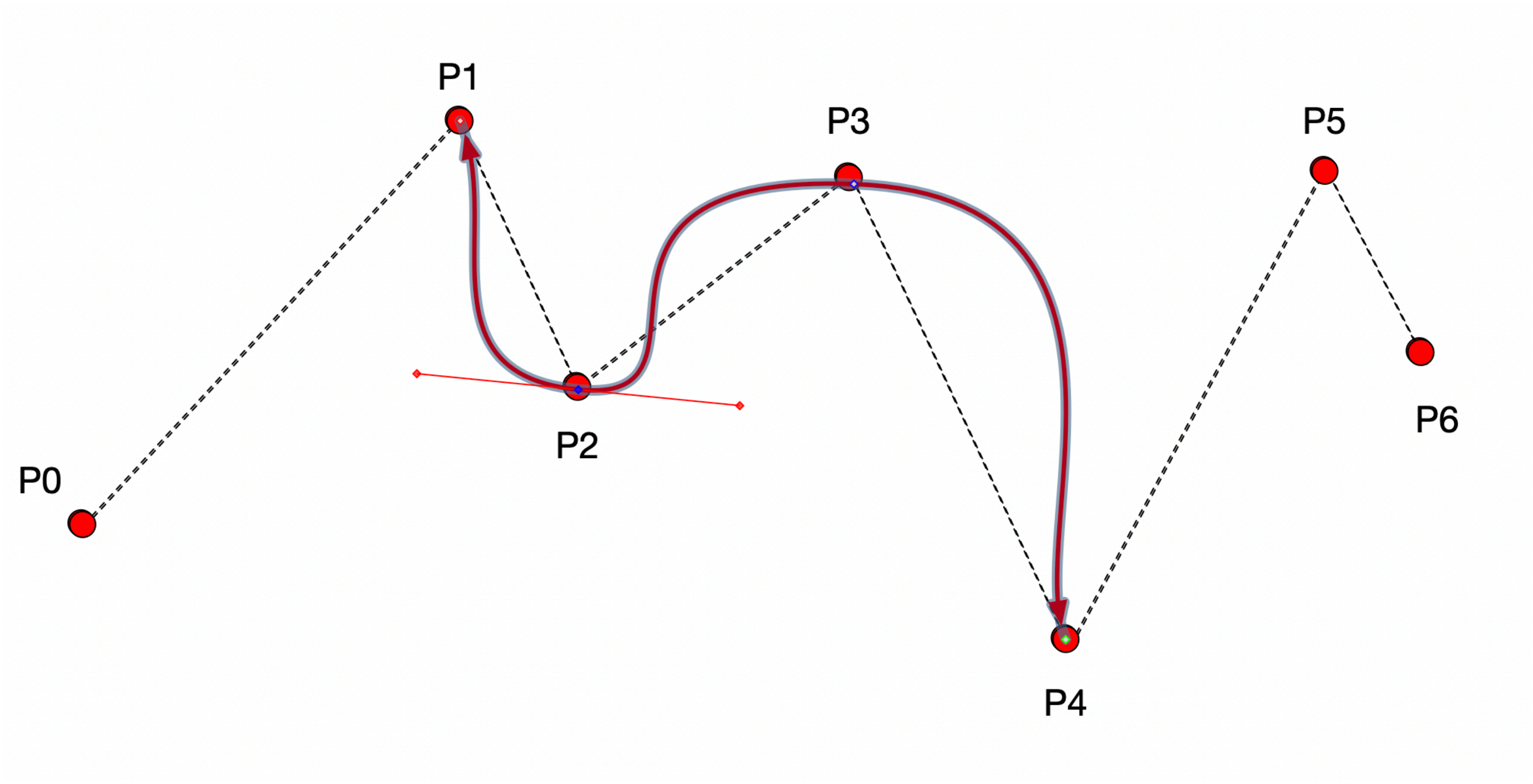
Calculating tangent at each point?

Problem – if we use vector to next point we don't get C1 continuity



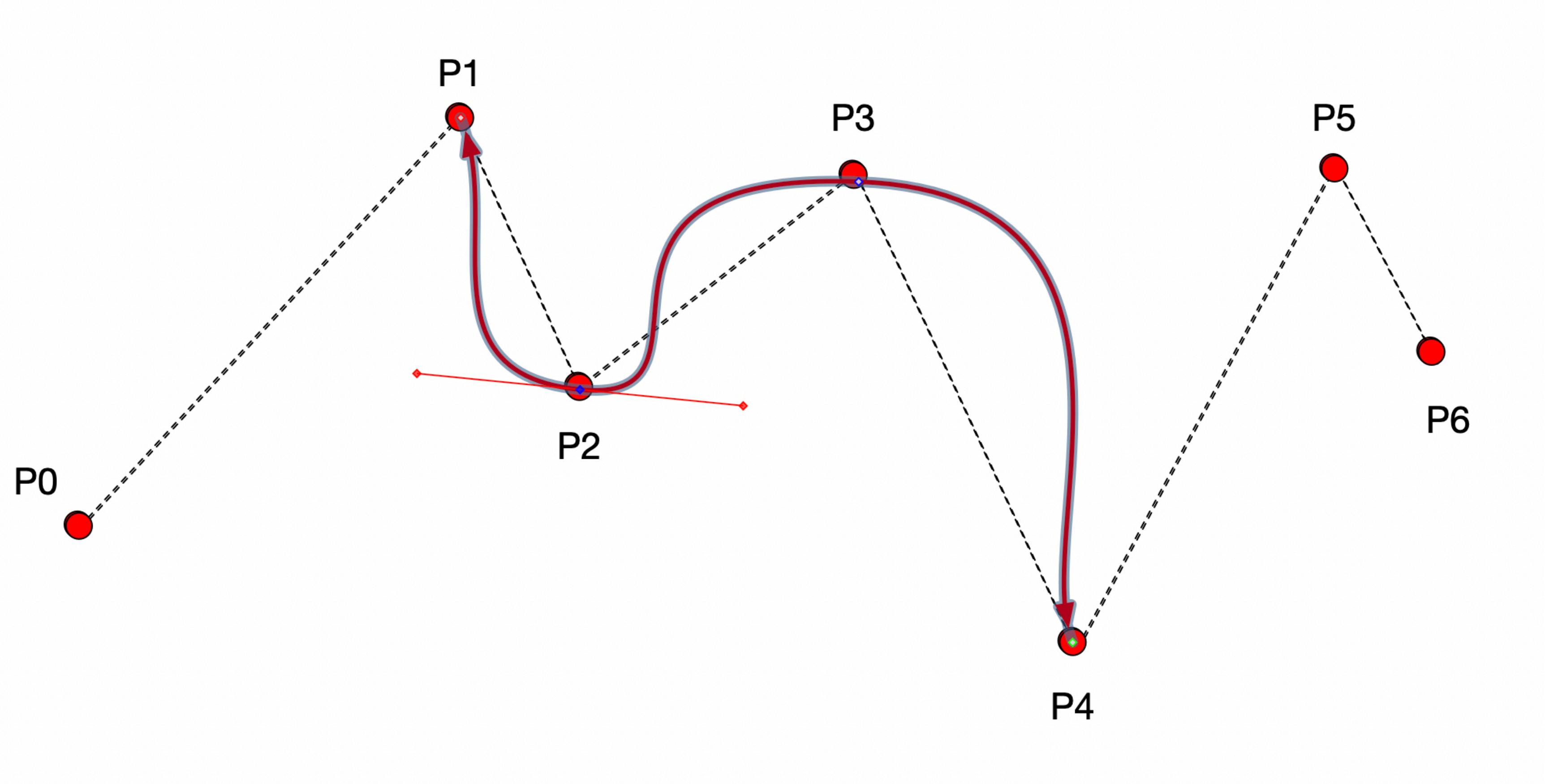
Solution: use vector P(i-1) to P(i+1)

Use same vector at P2 for segments P1 to P2, and P2 to P3

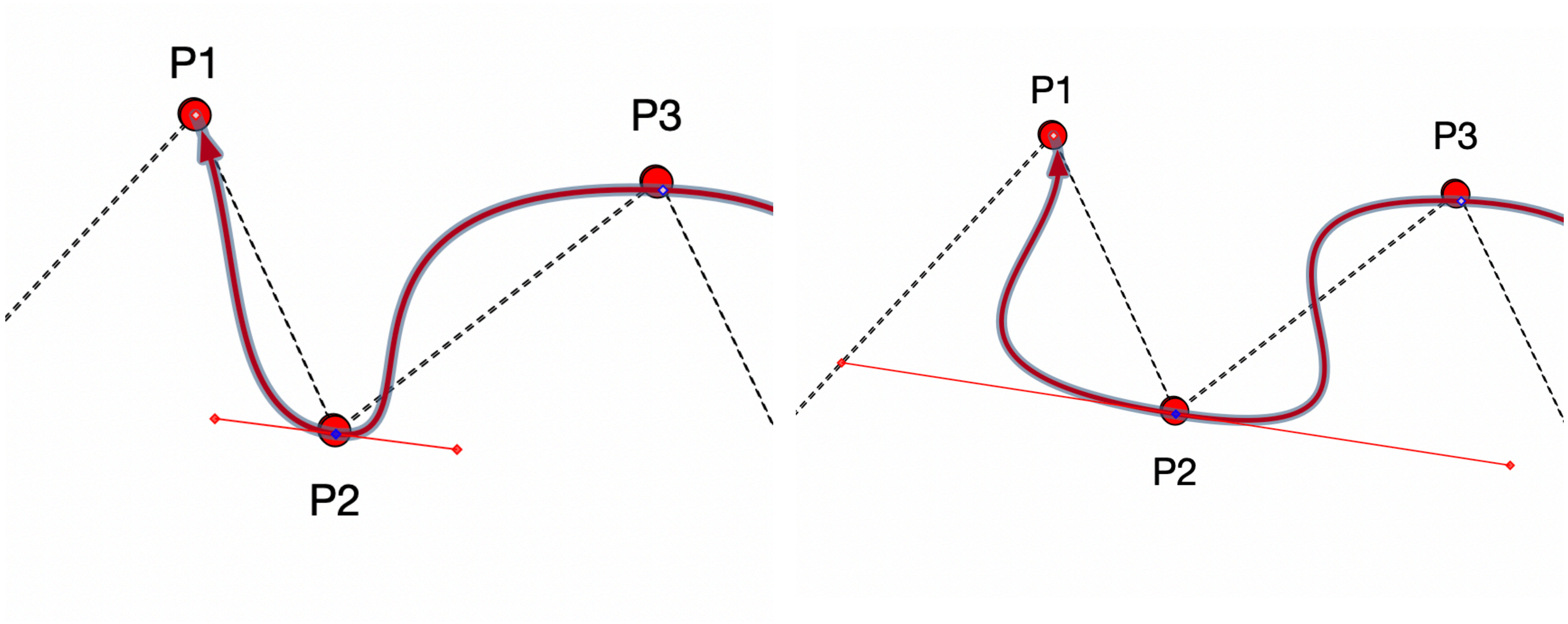


Solution: use vector P(i-1) to P(i+1)

Use same vector at P2 for segments P1 to P2, and P2 to P3

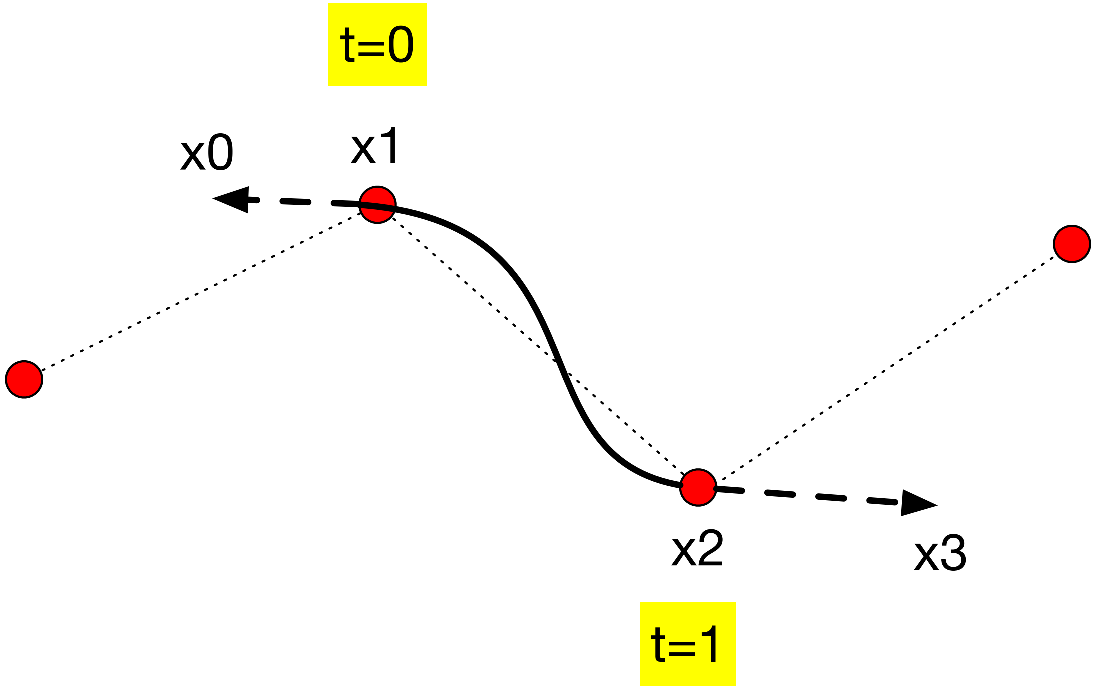


Scaling tangent => tightness of curve



Finding cubic coefficients a,b,c,d for segment

- $P_x(t) = at^3 + bt^2 + ct + d$
- $P'_x(t) = 4at^3 + 3bt + c$

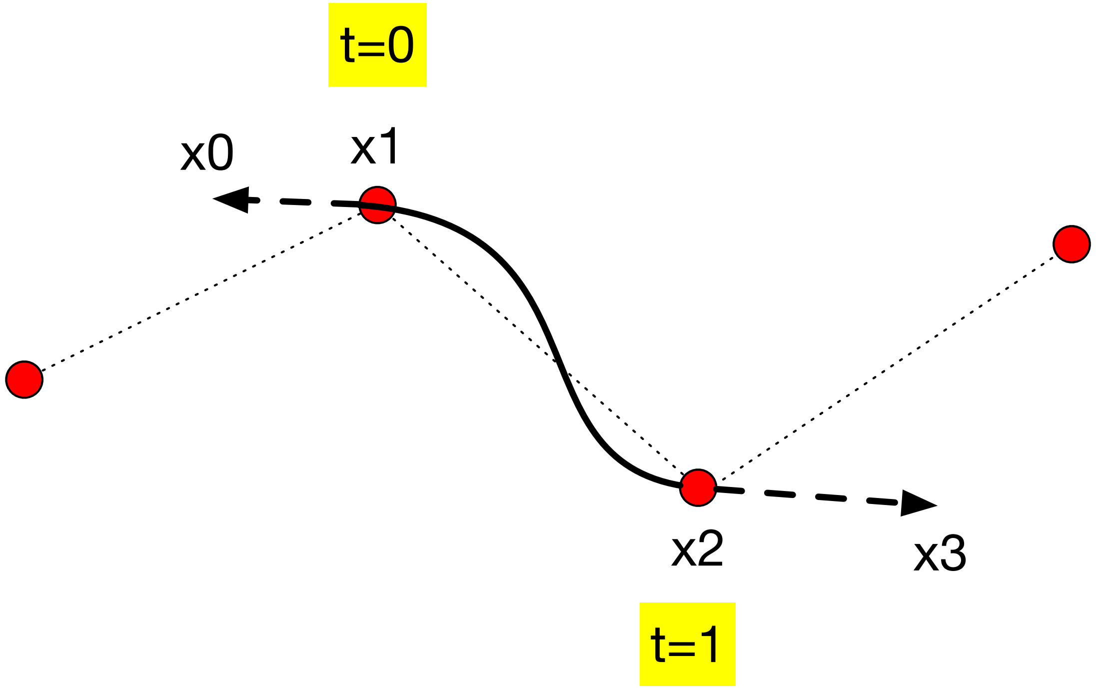


Finding cubic coefficients a,b,c,d for segment

- $P_x(t) = at^3 + bt^2 + ct + d$
- $P'_x(t) = 3at^2 + 2bt + c$

- $P_x(0) = x1 = \ d$
- $P'_x(0) = x0 = \ c$

- $P_x(1) = x2 = a + b + c + d$
- $P'_x(1) = x3 = 3a + 2b + c$



System of equations in four unknowns

- $x1 = d$
- $x0 = c$
- $x2 = a + b + c + d$
- $x3 = 3a + 2b + c$
- $\begin{bmatrix} x0 \\ x1 \\ x2 \\ x3 \end{bmatrix} = \begin{bmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 1 & 1 & 1 & 1 \\ 3 & 2 & 1 & 0 \end{bmatrix} \begin{bmatrix} a \\ b \\ c \\ d \end{bmatrix}$

- Solve?

System of equations in four unknowns

- $x1 = \ d$
- $x0 = \ c$
- $x2 = a + b + c + d$
- $x3 = 3a + 2b + c$

- $\begin{bmatrix} x0 \\ x1 \\ x2 \\ x3 \end{bmatrix} = \begin{bmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 1 & 1 & 1 & 1 \\ 3 & 2 & 1 & 0 \end{bmatrix} \begin{bmatrix} a \\ b \\ c \\ d \end{bmatrix}$

- Solve? M inverse

- $\begin{bmatrix} a \\ b \\ c \\ d \end{bmatrix} = \begin{bmatrix} 1 & 2 & -2 & 1 \\ -2 & -3 & 3 & -1 \\ 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \end{bmatrix} \begin{bmatrix} x0 \\ x1 \\ x2 \\ x3 \end{bmatrix}$

Using with Quadratic form   q^{T}Mp

- $P_x(t) = at^3 + bt^2 + ct + d$

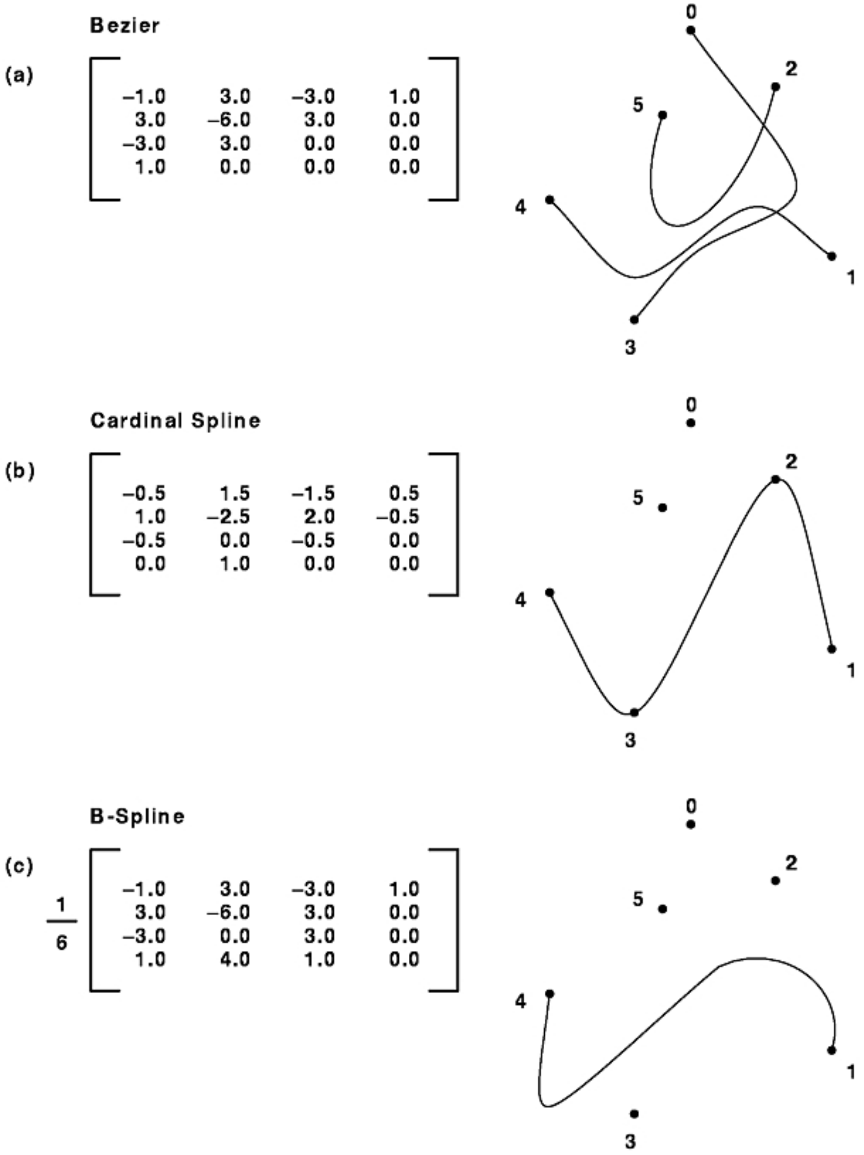
- $P_x(t) = [t^3 \quad t^2 \quad t \quad 1]\begin{bmatrix} 1 & 2 & -2 & 1 \\ -2 & -3 & 3 & -1 \\ 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \end{bmatrix}\begin{bmatrix} x0 \\ x1 \\ x2 \\ x3 \end{bmatrix}$

- $P(t) = [t^3 \quad t^2 \quad t \quad 1]\begin{bmatrix} 1 & 2 & -2 & 1 \\ -2 & -3 & 3 & -1 \\ 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \end{bmatrix}\begin{bmatrix} T0 \\ P1 \\ P2 \\ T3 \end{bmatrix}$

Spline equations

- Different constraints on points and tangents => different matrices
- Which one?

$$
\begin{aligned}
\mathbf{P}(t) &= \sum_{i=0}^{3} \mathbf{P}_i B_i(t) \\
&= (1-t)^3\mathbf{P}_0 + 3t(1-t)^2\mathbf{P}_1 + 3t^2(1-t)\mathbf{P}_2 + t^3\mathbf{P}_3 \\
&= \begin{bmatrix} (1-t)^3 & 3t(1-t)^2 & 3t^2(1-t) & t^3 \end{bmatrix} \begin{bmatrix} \mathbf{P}_0 \\ \mathbf{P}_1 \\ \mathbf{P}_2 \\ \mathbf{P}_3 \end{bmatrix} \\
&= \begin{bmatrix} 1 & t & t^2 & t^3 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & 0 \\ -3 & 3 & 0 & 0 \\ 3 & -6 & 3 & 0 \\ -1 & 3 & -3 & 1 \end{bmatrix} \begin{bmatrix} \mathbf{P}_0 \\ \mathbf{P}_1 \\ \mathbf{P}_2 \\ \mathbf{P}_3 \end{bmatrix}
\end{aligned}
$$

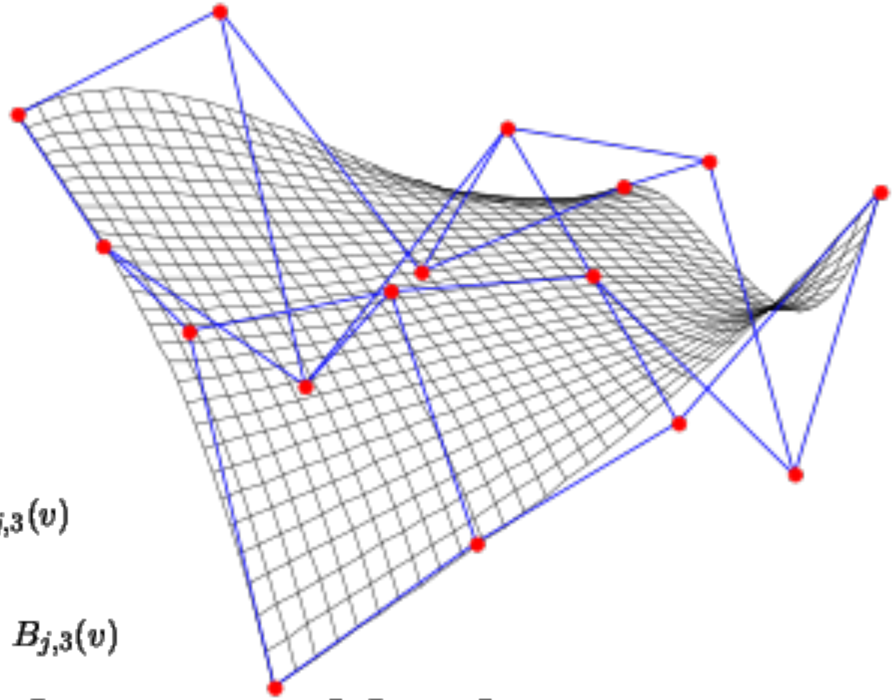


Bezier, Cardinal, and B-Spline Curves

Summary

- Take control points
- Compute curve/surface coefficients
  - Represent as matrix
- Draw parametrized curve/surface

- Bezier patch 16 control points



$$\mathbf{P}(u,v) = \sum_{j=0}^{3}\sum_{i=0}^{3}\mathbf{P}_{i,j}B_{i,3}(u)B_{j,3}(v)$$
$$= \sum_{j=0}^{3}\left[\sum_{i=0}^{3}\mathbf{P}_{i,j}B_{i,3}(u)\right]B_{j,3}(v)$$
$$= \sum_{j=0}^{3}\begin{bmatrix}1 & u & u^2 & u^3\end{bmatrix}\begin{bmatrix}1 & 0 & 0 & 0\\ -3 & 3 & 0 & 0\\ 3 & -6 & 3 & 0\\ -1 & 3 & -3 & 1\end{bmatrix}\begin{bmatrix}\mathbf{P}_{0,j}\\ \mathbf{P}_{1,j}\\ \mathbf{P}_{2,j}\\ \mathbf{P}_{3,j}\end{bmatrix}B_{j,3}(v)$$
$$= \begin{bmatrix}1 & u & u^2 & u^3\end{bmatrix}\begin{bmatrix}1 & 0 & 0 & 0\\ -3 & 3 & 0 & 0\\ 3 & -6 & 3 & 0\\ -1 & 3 & -3 & 1\end{bmatrix}\begin{bmatrix}\mathbf{P}_{0,0} & \mathbf{P}_{0,1} & \mathbf{P}_{0,2} & \mathbf{P}_{0,3}\\ \mathbf{P}_{1,0} & \mathbf{P}_{1,1} & \mathbf{P}_{1,2} & \mathbf{P}_{1,3}\\ \mathbf{P}_{2,0} & \mathbf{P}_{2,1} & \mathbf{P}_{2,2} & \mathbf{P}_{2,3}\\ \mathbf{P}_{3,0} & \mathbf{P}_{3,1} & \mathbf{P}_{3,2} & \mathbf{P}_{3,3}\end{bmatrix}\begin{bmatrix}1 & -3 & 3 & -1\\ 0 & 3 & -6 & 3\\ 0 & 0 & 3 & -3\\ 0 & 0 & 0 & 1\end{bmatrix}\begin{bmatrix}1\\ v\\ v^2\\ v^3\end{bmatrix}$$

Crowd motion

- How do multiple agents move without colliding?
- Detect collisions in advance

Crowd motion

- How do multiple agents move without colliding?
- Detect collisions in advance
- Dynamic obstacles – anticipate where someone will be

Crowd motion

- Model each agent with

- Current position $P_i(0)$
- Current velocity $\vec{v}_i(0)$
- Target velocity $\vec{v}_i{}^0(0)$
  - Towards goal

- Forces $\vec{F}_i(0)$ push on agent

Possible forces

- Like boid flocking?

Possible forces

- Like boid flocking?

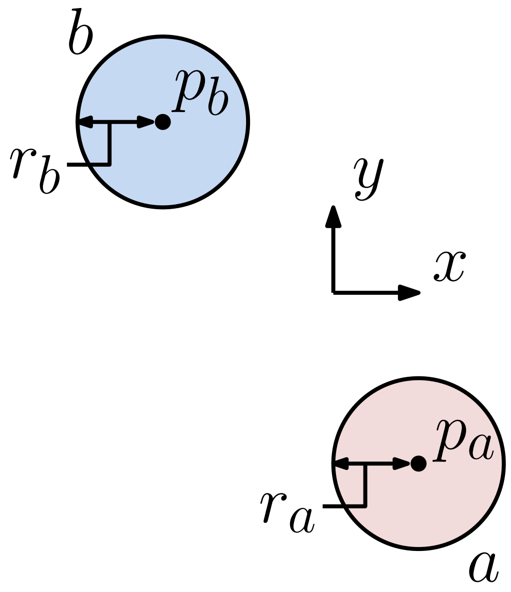
- Separation
- Obstacle Avoidance
- Attraction
- Traffic signals and social conventions
- Individual variations

Velocity obstacles

- Compute forbidden velocities
  - That would lead to collision

- Example
  - Agent **a** walking towards obstacle **b**

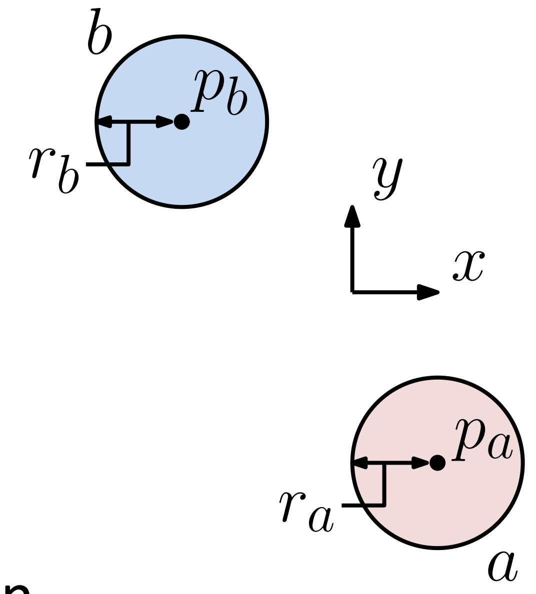
  - What velocities at time i cause collision?



Velocity obstacles

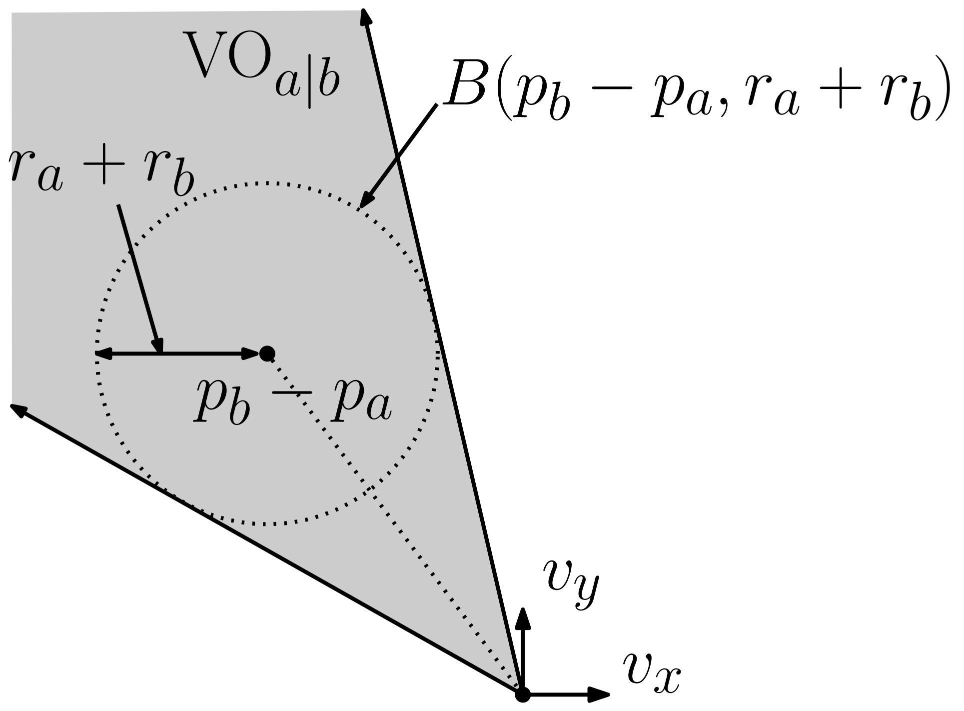
- Compute forbidden velocities
  - That would lead to collision

- Example
  - Agent **a** walking towards obstacle **b**
  - What velocities at time i cause collision?
  - Velocity v = (pb-pa) causes immediate collision
  - Others?



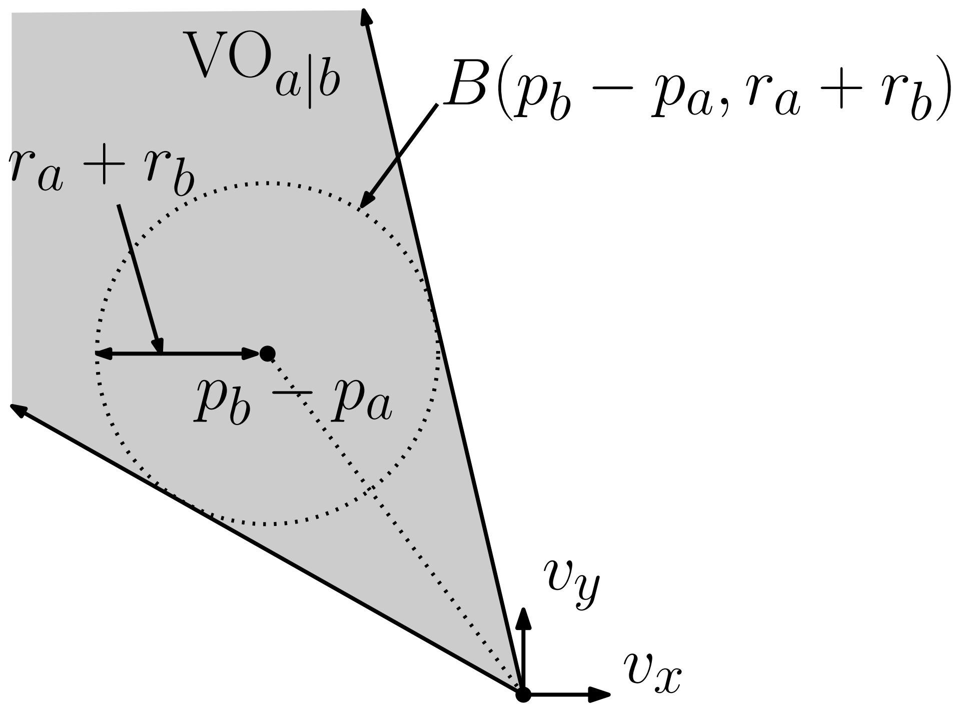
Velocity obstacles

- Region $V0_{(a|b)}$ of forbidden velocities
- Cone around Ball B(pb-pa, ra+rb)



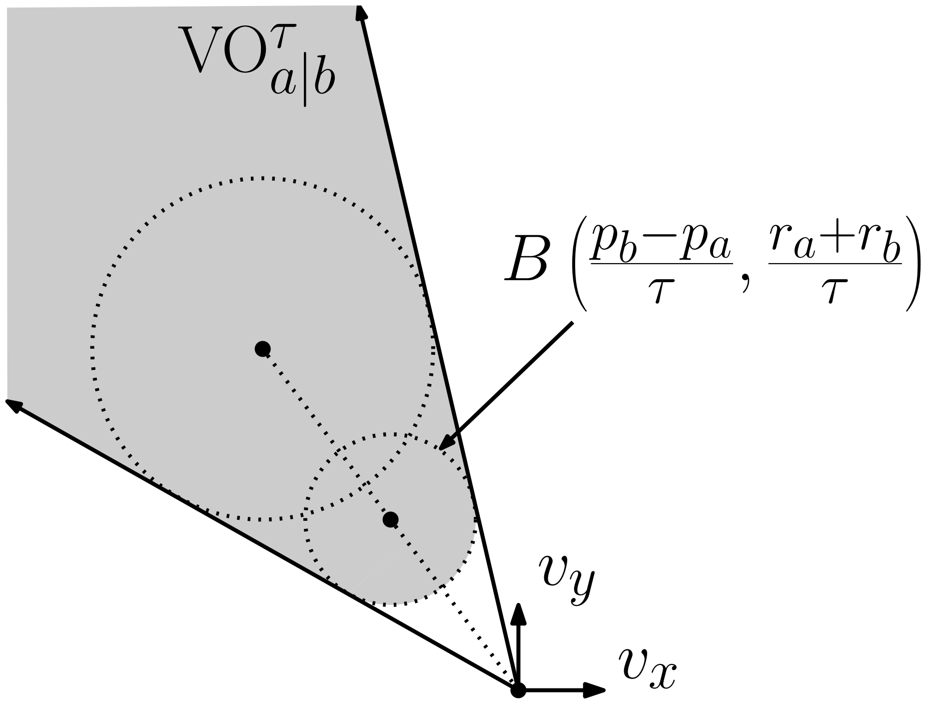
Velocity obstacles

- Region $V0_{(a|b)}$ of forbidden velocities
- Cone around Ball B(pb-pa, ra+rb)
- Apex of cone is what?

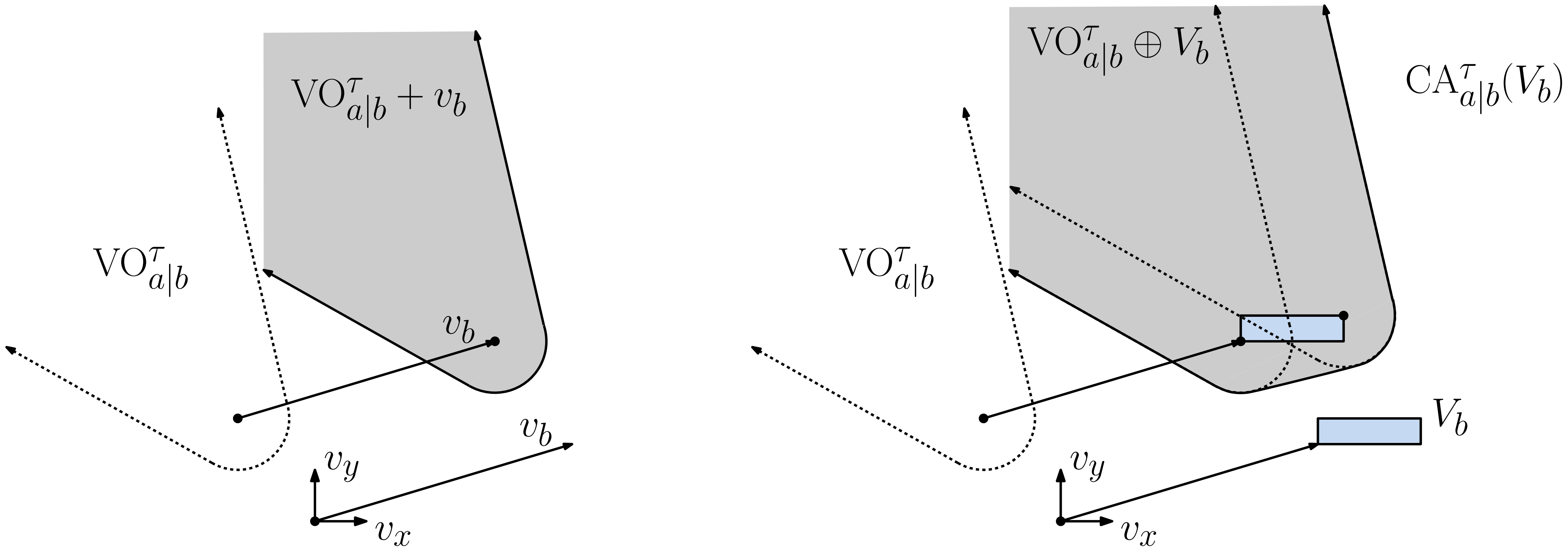


Velocity obstacles

- Region V0$_{(a|b)}$ of forbidden velocities
- Apex of cone is very slow velocities
- Limit length of future time to (0,tau)
- Limiting time truncates cone – why?



Obstacle **b** moving?

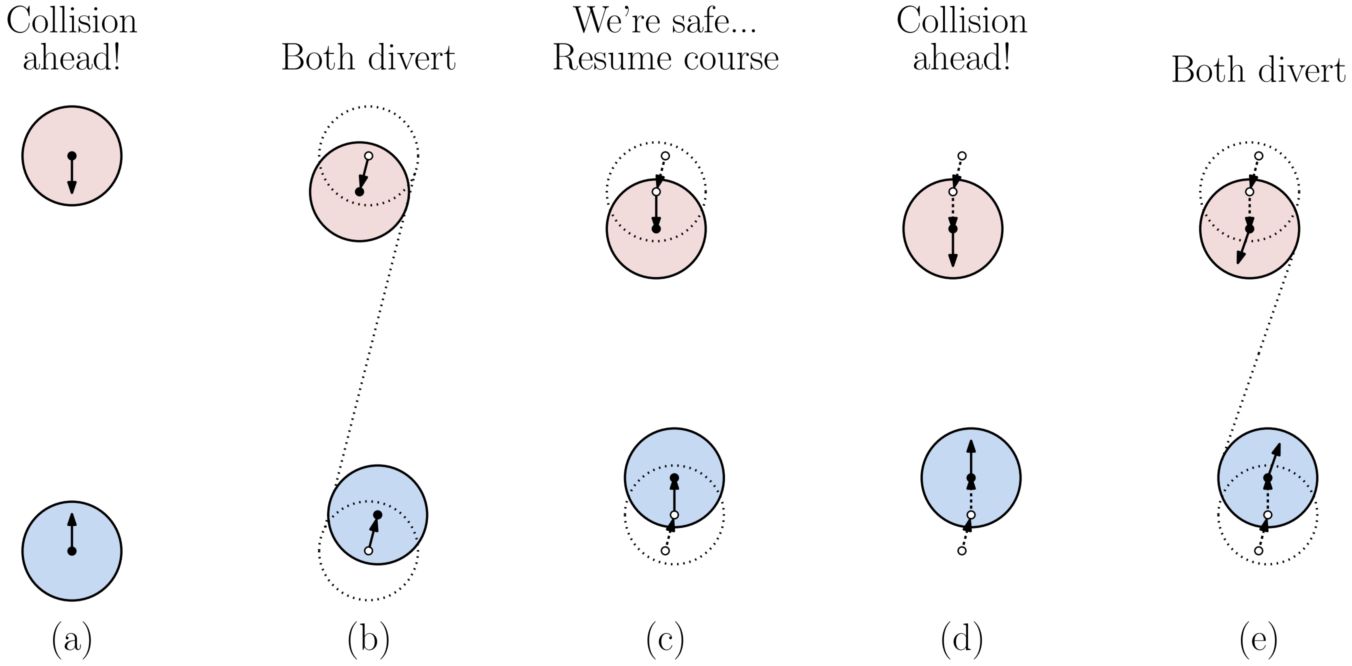


(a) (b)

Who is responsible for avoiding collision?

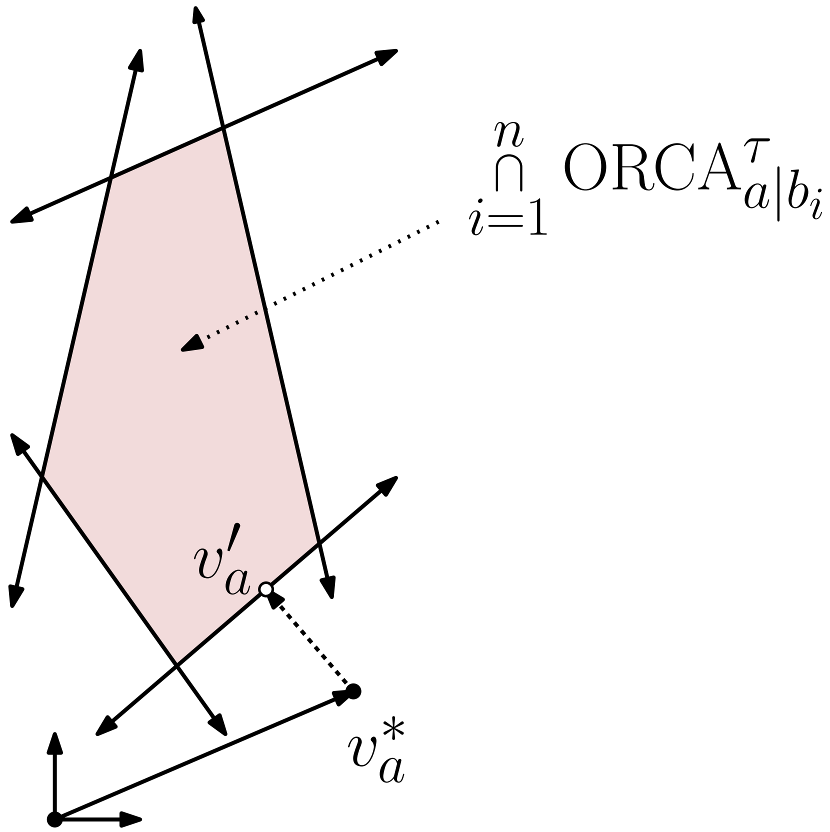
- Both agents fully responsible
- Oscillating motion

- Other avoids
- Your path becomes clear
- You resume original path
- Collision!



Avoiding multiple agents

- Simplify each agent's forbidden velocity region to half plane
- Intersect acceptable velocity regions to get polygon
- Take velocity v'a nearest to target velocity v*a



Lin and Manocha

- https://www.youtube.com/watch?v=lyyyEcy_9so
- https://www.youtube.com/watch?v=xme4pRelwJ0

Problem 3. (20 points) Consider the collection of shaded rectangular obstacles shown in the figure below, all contained within a large enclosing rectangle. Also, consider the triangular robot, whose reference point is located at a point s. (You may take s to be the origin.)

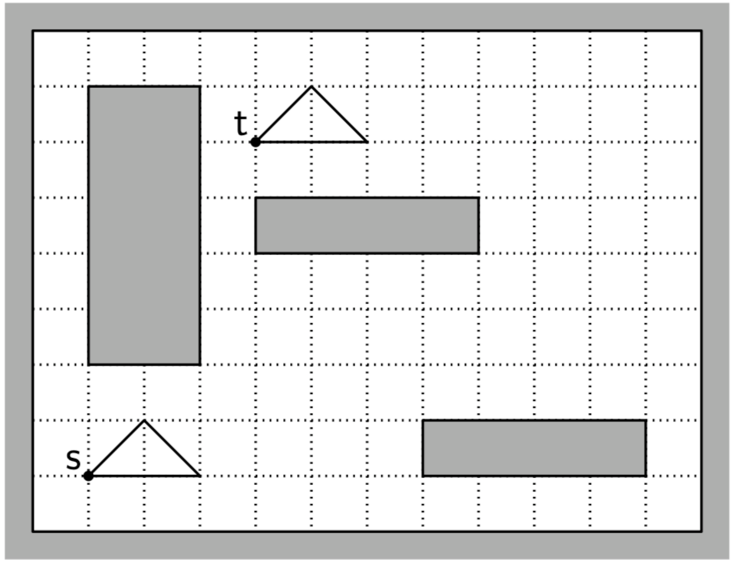


Figure 2: Problem 3.

(a) Draw the C-obstacles for the three rectangular obstacles, including the C-obstacle from region lying outside the large enclosing rectangle.

(b) Either draw an obstacle-avoiding path for the robot from s to t, or explain why it doesn't exist.