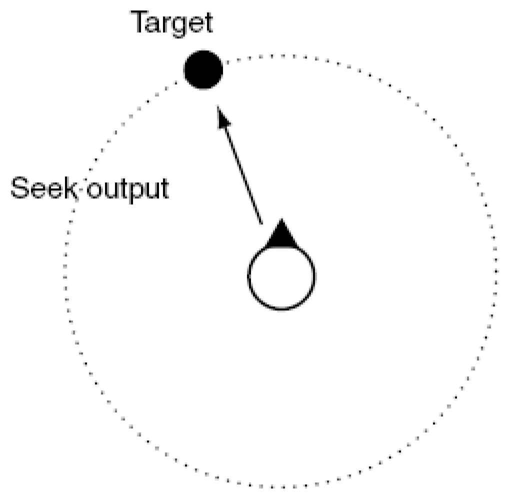
Movement: Behaviors, Crowds

Dynamic Wander Behavior

- Move towards a random target

compute motion direction

kinematic wander:



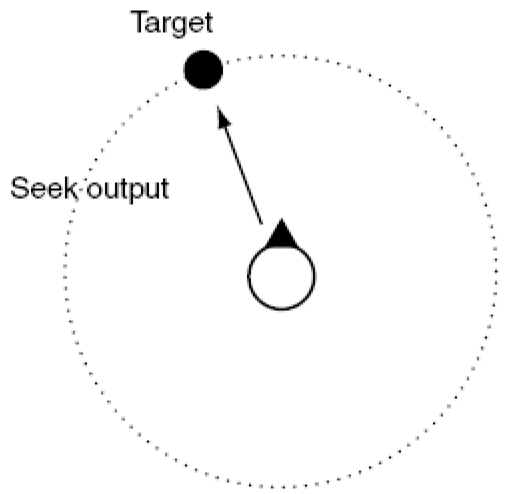
Unrealistic effects?

Dynamic Wander Behavior

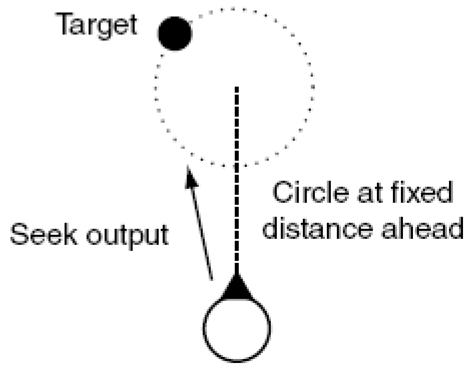
- Move towards a random target

compute motion direction

kinematic wander:



full dynamic wander:



from "Artificial Intelligence for Games" by I. Millington & J. y ge

Dynamic Wander Behavior

compute motion direction

- Move towards a random target

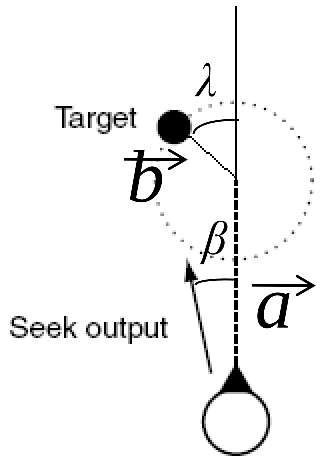
  λ = *random with bias towards 0*

  $target = P + \vec{a} + \vec{b}$

  $A = max.acceleration * normalize(target)$

  $dd\Psi = K\beta$ *limited by max. angular acceleration*

full dynamic wander:



from "Artificial Intelligence for Games" by I. Millington & J. y ge

Dynamic Wander Behavior

compute motion direction

$\downarrow$

- Move towards a random target

  λ = random with bias towards 0

  $target = P + \vec{a} + \vec{b}$

  *A=max.acceleration*normalize(target)*

  $dd\Psi = K\beta$ limited by max. angular acceleration

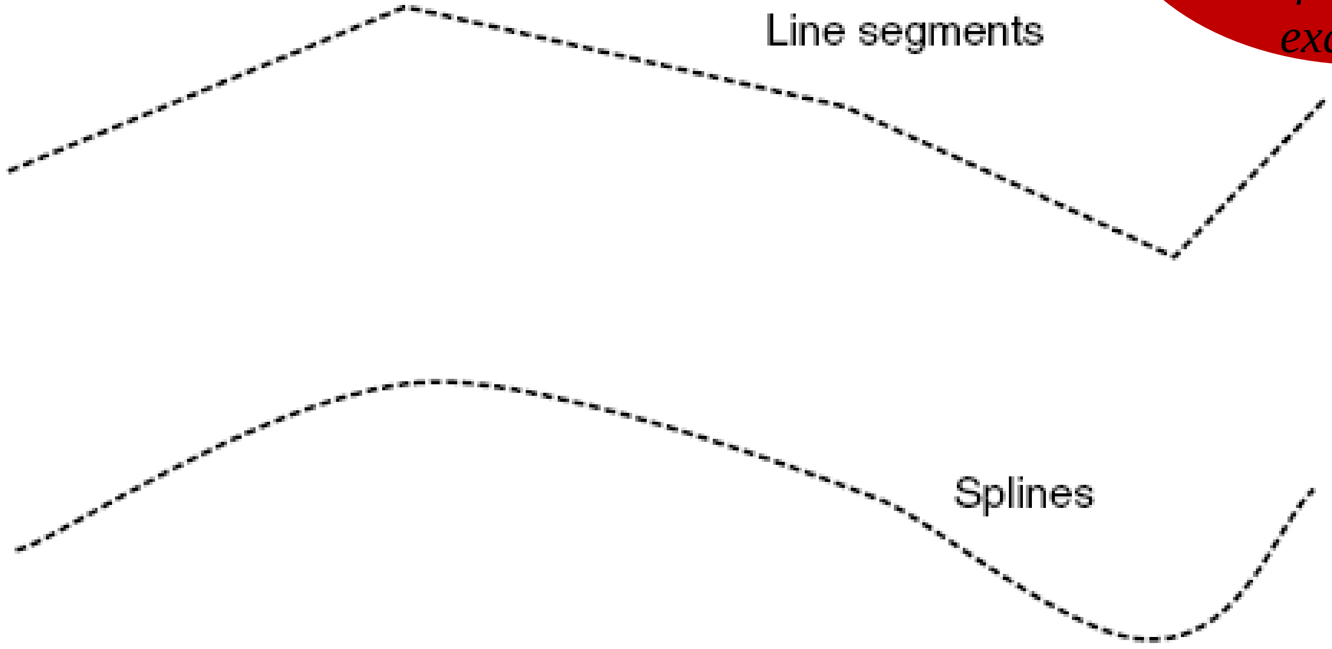
http://www.red3d.com/cwr/steer/Wander.html

Path Following

- Follow a path given by a series of line segments or splines

compute motion direction

Why need a path following algorithm (and not follow path exactly)?



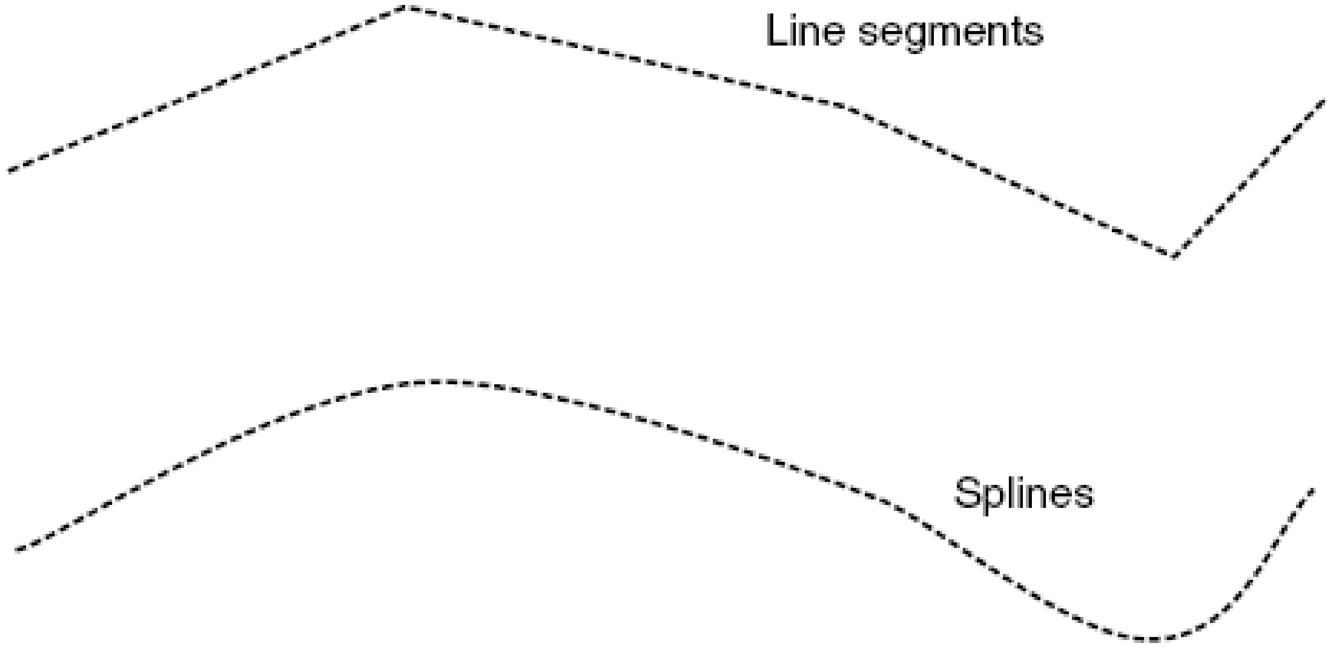
from "Artificial Intelligence for Games" by I. Millington & J. y ge

Path Following

• Follow a path given by a series of line segments or splines

compute motion direction

Any ideas for how to do it?



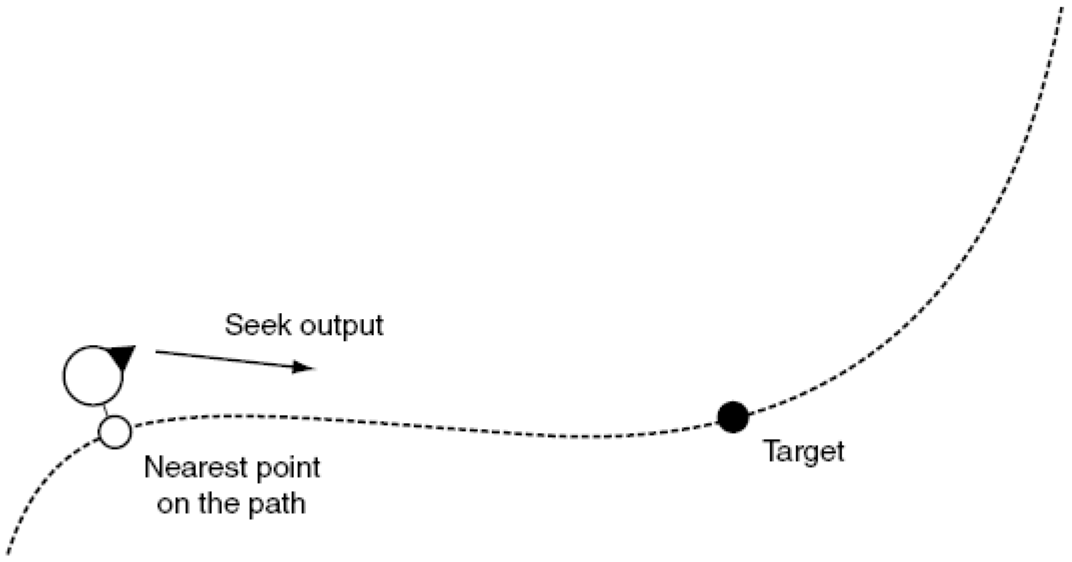
from "Artificial Intelligence for Games" by I. Millington & J. y ge

Path Following

- Follow a path given by a series of line segments or splines

compute motion direction

```
compute nearest point $P_{near}$ on the path
target=$P_{near}$ +offset by distance (time) L along the path
execute seek(target)
```



from “Artificial Intelligence for Games” by I. Millington & J. y ge

Path Following

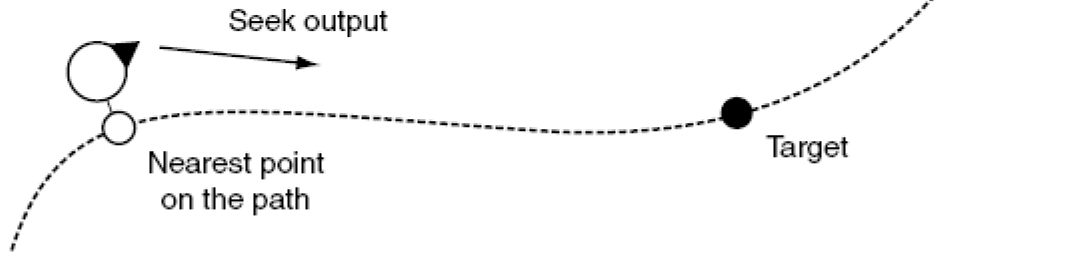
- Follow a path given by a series of line segments or splines

compute motion direction

```
 compute nearest point $P_{near}$ on the path
target=$P_{near}$ +offset by distance (time) L along the path
execute seek(target)
```

How to find a nearest point?

Any issues?



from "Artificial Intelligence for Games" by I. Millington & J. y ge

Path Following

compute motion direction

- Follow a path given by a series of line segments or splines

compute nearest point P_{near} on the path

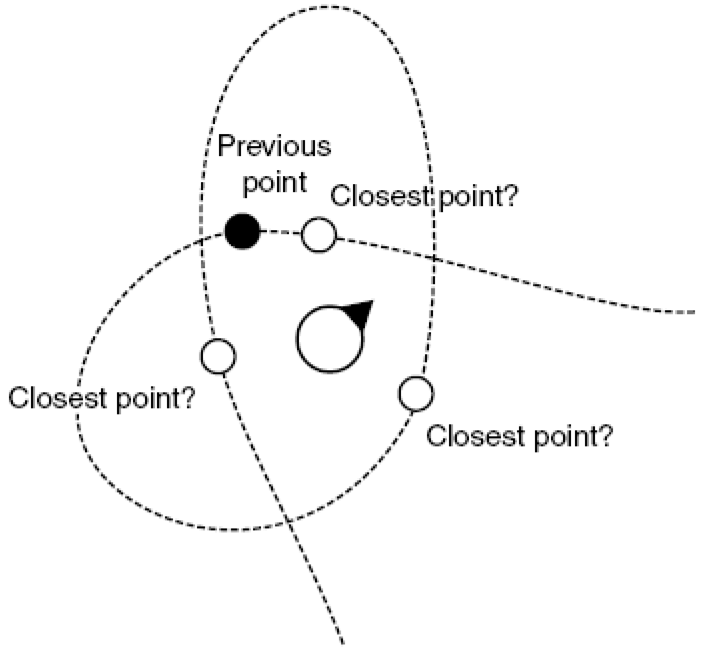
target=P_{near} +offset by distance (time) L along the path

execute seek(target)

How to find a nearest point?

Any issues?

Any solutions?



from "Artificial Intelligence for Games" by I. Millington & J. y ge

Path Following

- Follow a path given by a series of line segments or splines

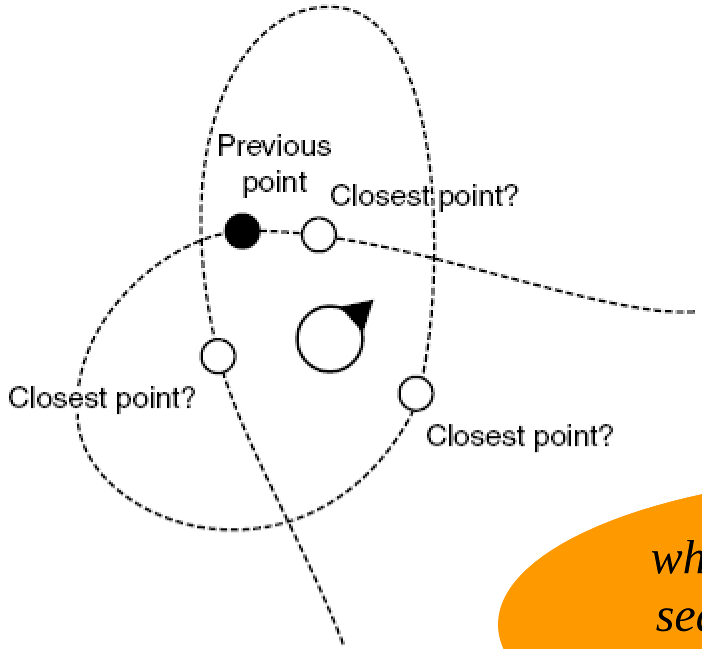
compute motion direction

compute nearest point P_{near} on the path
target=P_{near} +offset by distance (time) L along the path
execute seek(target)

How to find a nearest point?

Any issues?

Any solutions?



when computing P_{near}, only search small path segment in front of previous P_{near}

from "Artificial Intelligence for Games" by

Path Following

- Follow a path given by a series of line segments or splines

compute motion direction

compute nearest point P_{near} on the path
target=P_{near} +offset by distance (time) L along the path
execute seek(target)

http://www.red3d.com/cwr/steer/PathFollow.html

Maintain Separation

compute motion direction

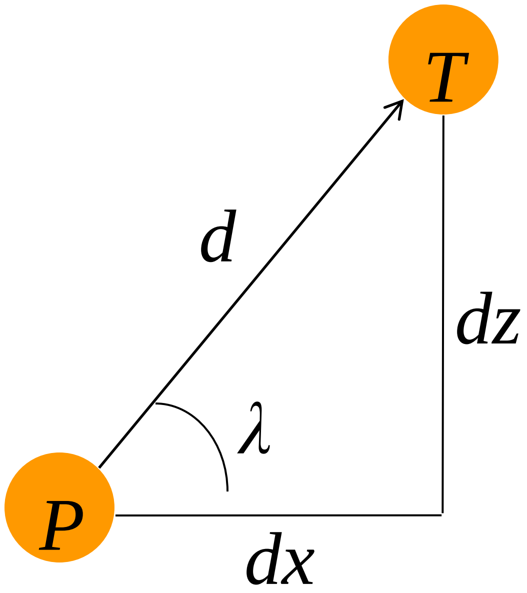
- Maintain distance from nearby characters

    for all nearby characters T

    strength = min(K/d², max. accel)

    *A=-strength *normalize([dx,dz])*

    $dd\Psi = K_1(\Psi - \lambda - \pi)$ *limited by max. angular acceleration*



You dance with Keeg Gibn.
You dance with Keeg Gibn.
You dance with Keeg Gibn.
[1. General] Beryl: get down tonight?
Dunmore says: join in folks
[1. General] FourInchTool: Considering we're dwarves, little is right.
Damador bursts into dance.
Dunmore bursts into dance.

TigerLibra gains Devotion Aura.
BeoWolven gains Devotion Aura.
Devotion Aura fades from BeoWolven.
Devotion Aura fades from BaleogFierce.
Devotion Aura fades from TigerLibra.
Devotion Aura fades from Erebuni.
Devotion Aura fades from Brunhilde.
Devotion Aura fades from BaleogFierce.

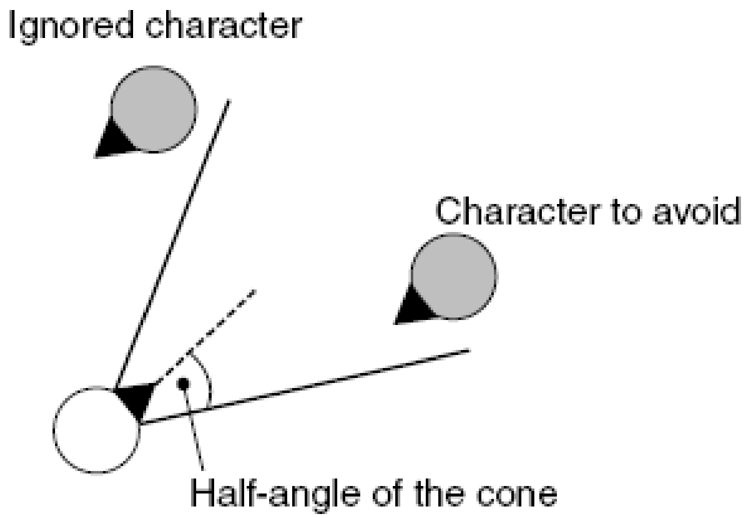
Damador
Dwarf Paladin
Level 8

Collision Avoidance using Separation

- Avoid collisions

for all characters T within cone of view
run separation behavior

compute motion direction



Any issues?

from "Artificial Intelligence for Games" by I. Millington & J. y ge

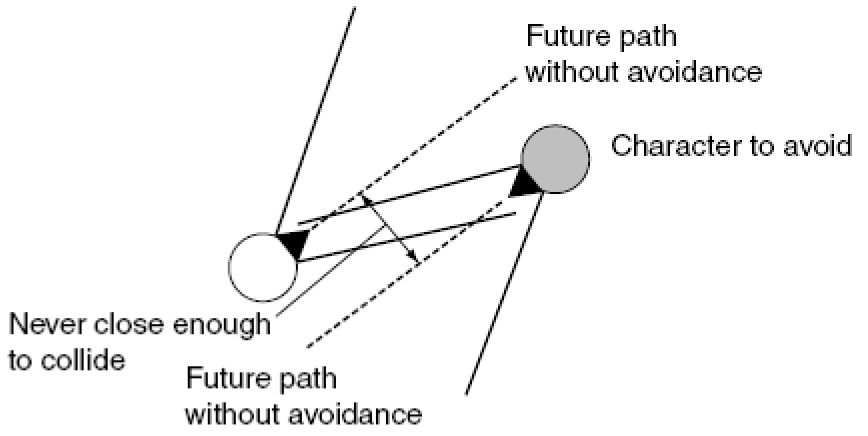
Collision Avoidance using Separation

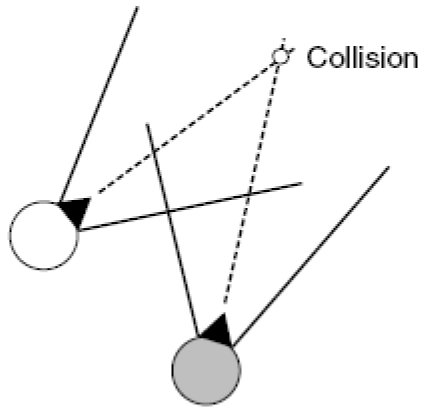
- Avoid collisions

for all characters T within cone of view

run separation behavior

compute motion direction


Future path without avoidance
Character to avoid
Never close enough to collide
Future path without avoidance



Any issues?

Any solutions?

from "Artificial Intelligence for Games" by I. Millington & J. y ge

Collision Avoidance with Collision Prediction

- Avoid collisions

for all characters with small $t_{closest}$

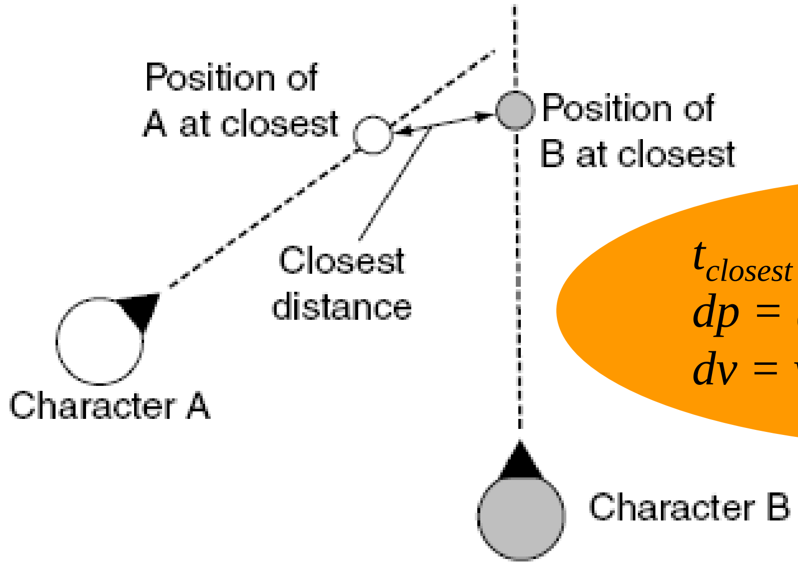
$$B_{closest} = B_{current} + v_B * t_{closest}$$

$$A_{closest} = A_{current} + v_A * t_{closest}$$

Flee as if character at $A_{closest}$ *and target at* $B_{closest}$

compute motion direction

What to do if many characters nearby?



$t_{closest} = -dp.dv/dv^2$, *where*
$dp = (B_{current} - A_{current})$
$dv = v_B - v_A$

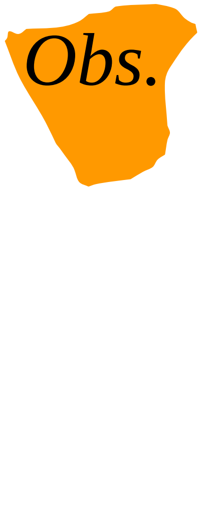
from "Artificial Intelligence for Games" by I. Millington & J. y ge

Obstacle Avoidance

- Avoiding collisions with obstacles

compute motion direction

Any ideas how to do it?



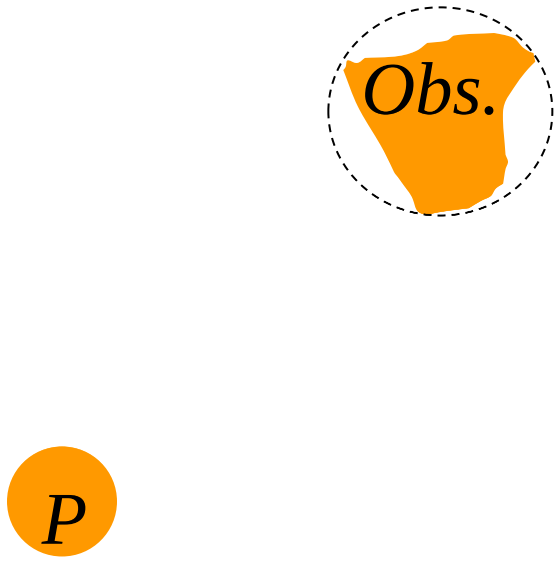
Obstacle Avoidance

- Avoiding collisions with obstacles

compute motion direction

for all obstacles that can be approximated with a circle

run separation behavior



Obstacle Avoidance

- Avoiding collisions with obstacles

```
for all obstacles that can be approximated with a circle
    run separation behavior
```

compute motion direction

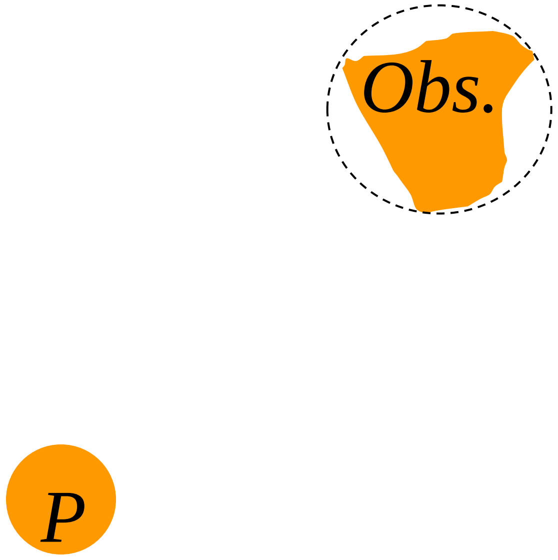
http://www.red3d.com/cwr/steer/Obstacle.html

Obstacle Avoidance

- Avoiding collisions with obstacles

compute motion direction

for all obstacles that can be approximated with a circle
run separation behavior



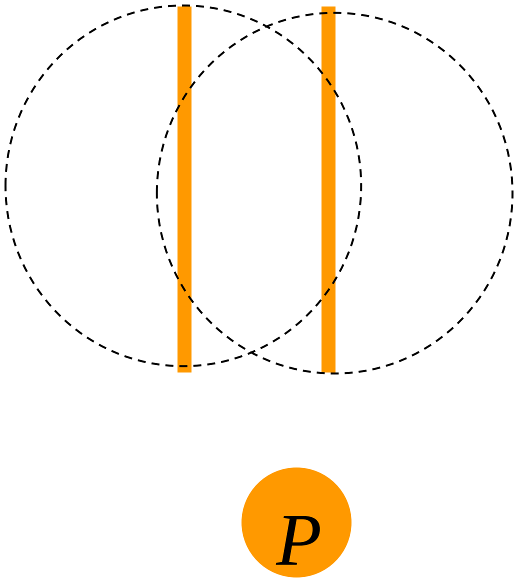
Failure examples with non-circular obstacles?

Obstacle Avoidance

- Avoiding collisions with obstacles

compute motion direction

for all obstacles that can be approximated with a circle
run separation behavior



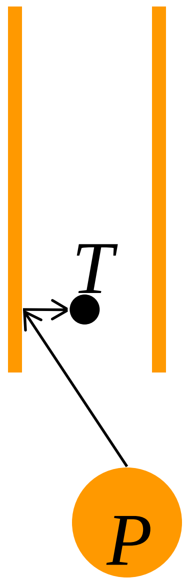
Failure examples with non-circular obstacles?

Obstacle Avoidance

- Avoiding collisions with wall-like obstacles

```
for all other (nearby) obstacles
    shoot a ray in the current motion direction
    find collision if any
    set target T to short distance along normal to
        collision surface
    seek on T
```

compute motion direction

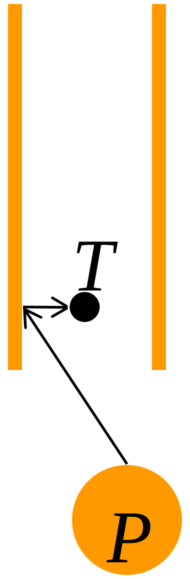


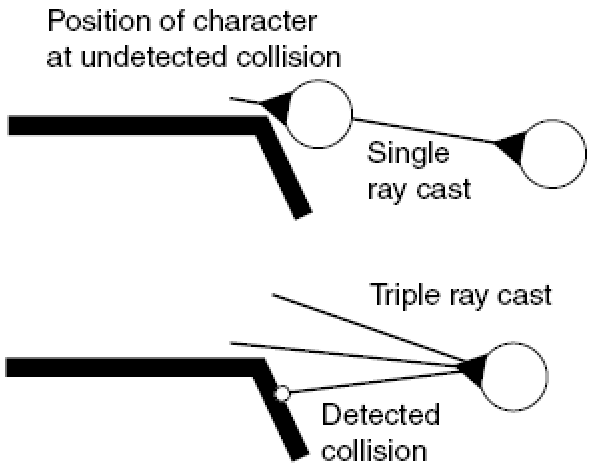
Obstacle Avoidance

- Avoiding collisions with wall-like obstacles

compute motion direction

```
for all other (nearby) obstacles
    shoot rays in the current motion direction
    find collision if any
    set target T to short distance along normal to
        collision surface
    seek on T
```





from "Artificial Intelligence for Games" by I. Millington & J. y

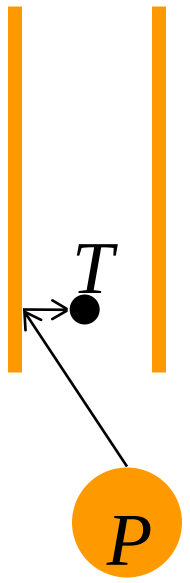
Obstacle Avoidance

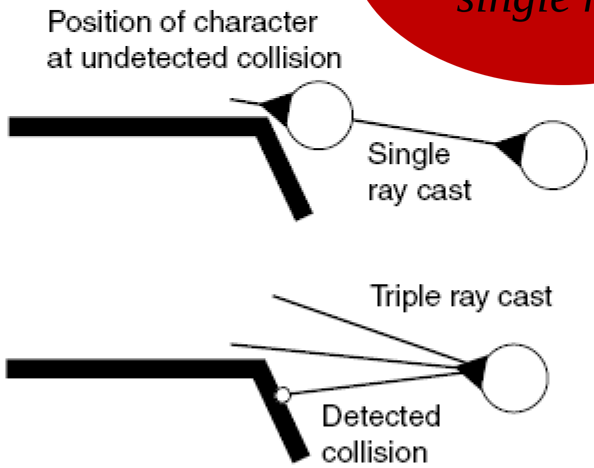
- Avoiding collisions with wall-like obstacles

```
for all other (nearby) obstacles
    shoot rays in the current motion direction
    find collision if any
    set target T to short distance along normal to
        collision surface
    seek on T
```

compute motion direction

What is the problem with single ray?





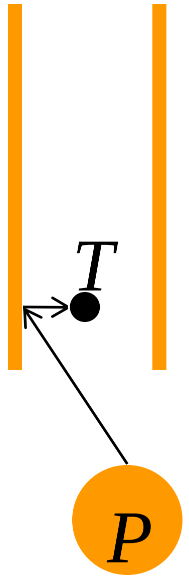
from "Artificial Intelligence for Games" by I. Millington & J. y

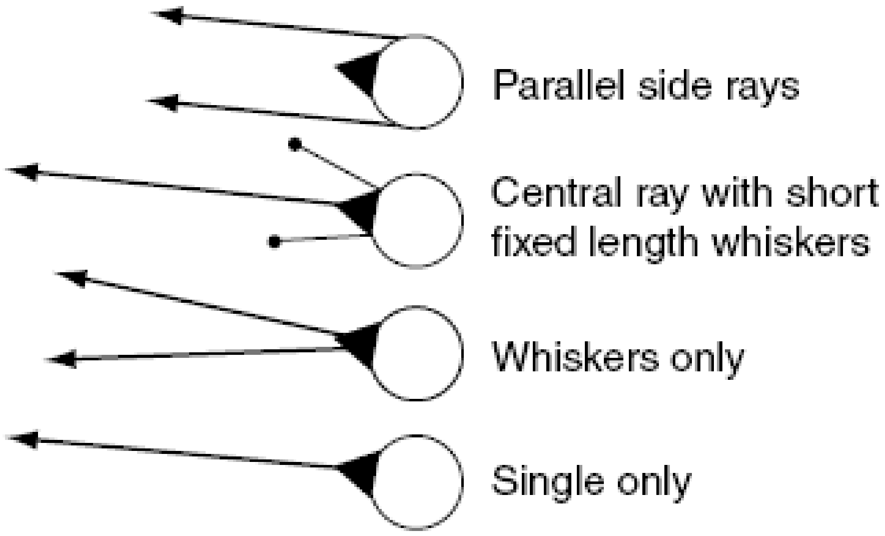
Obstacle Avoidance

- Avoiding collisions with wall-like obstacles

```
for all other (nearby) obstacles
    shoot rays in the current motion direction
    find collision if any
    set target T to short distance along normal to
        collision surface
    seek on T
```

compute motion direction





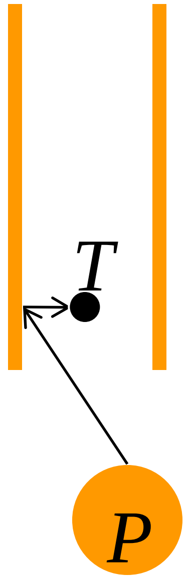
from "Artificial Intelligence for Games" by I. Millington & J. y

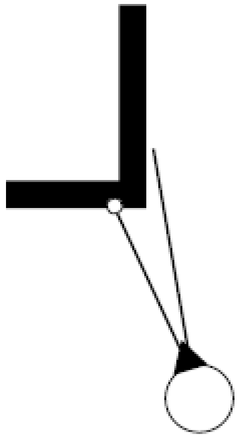
Obstacle Avoidance

- Avoiding collisions with wall-like obstacles

compute motion direction

```
for all other (nearby) obstacles
    shoot rays in the current motion direction
    find collision if any
    set target T to short distance along normal to
        collision surface
    seek on T
```





What will happen?

Solutions?

from "Artificial Intelligence for Games" by I. Millington & J. y

Obstacle Avoidance

- Avoiding collisions with wall-like obstacles

compute motion direction

```
for all other (nearby) obstacles
    shoot rays in the current motion direction
    find collision if any
    set target T to short distance along normal to
        collision surface
    seek on T
```

http://www.red3d.com/cwr/steer/Wall.html

Combining Behaviors to Get Complex Behaviors

compute motion direction

Follow Path

Avoid Obstacles

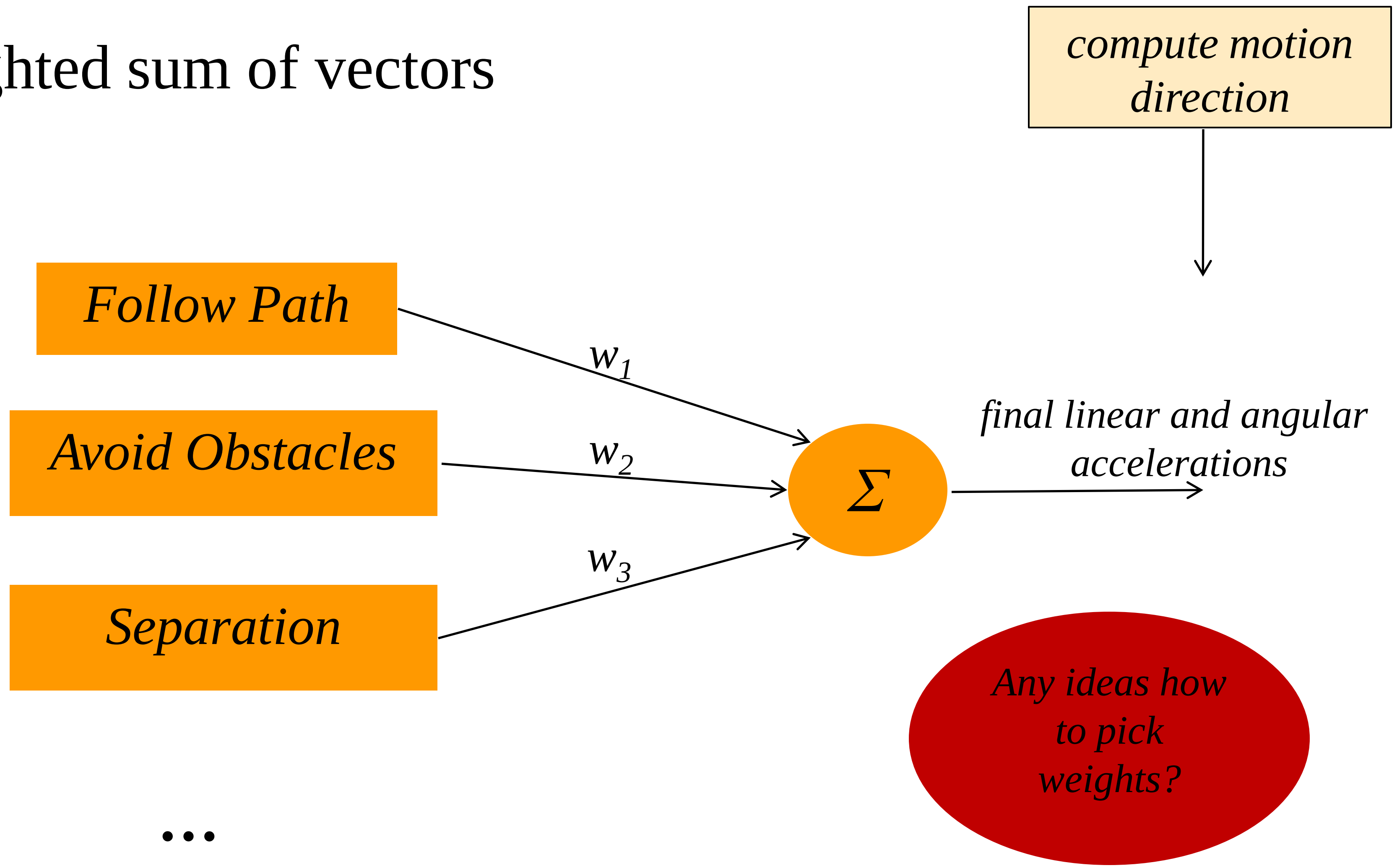
Separation

…

Any ideas how to combine them?

Combining Behaviors to Get Complex Behaviors

- Weighted sum of vectors

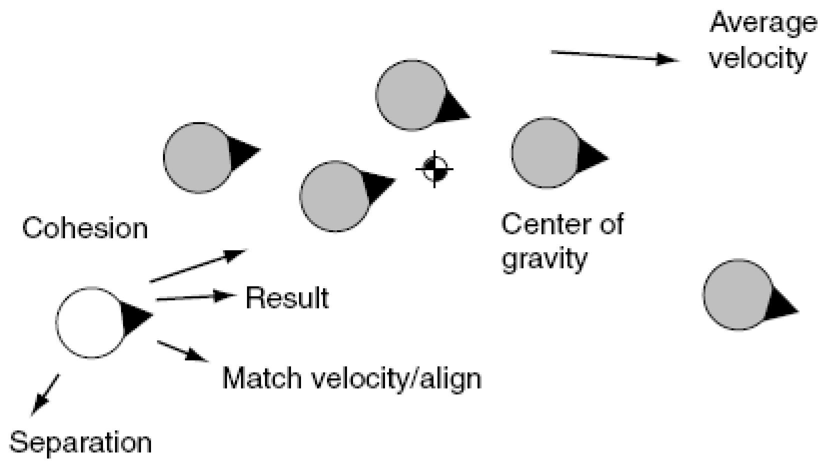


Combining Behaviors to Get Complex Behaviors

- Weighted sum of vectors

Flocking (Boids model):

$$\begin{tikzcd}
\textit{Cohesion} \arrow[dr, "w_1"] & & \textit{compute motion direction} \arrow[d] \\
\textit{Velocity Matching} \arrow[r, "w_2"] & \Sigma \arrow[r, "\textit{final linear and angular accelerations}"] & {} \\
\textit{Separation} \arrow[ur, "w_3"'] & &
\end{tikzcd}$$

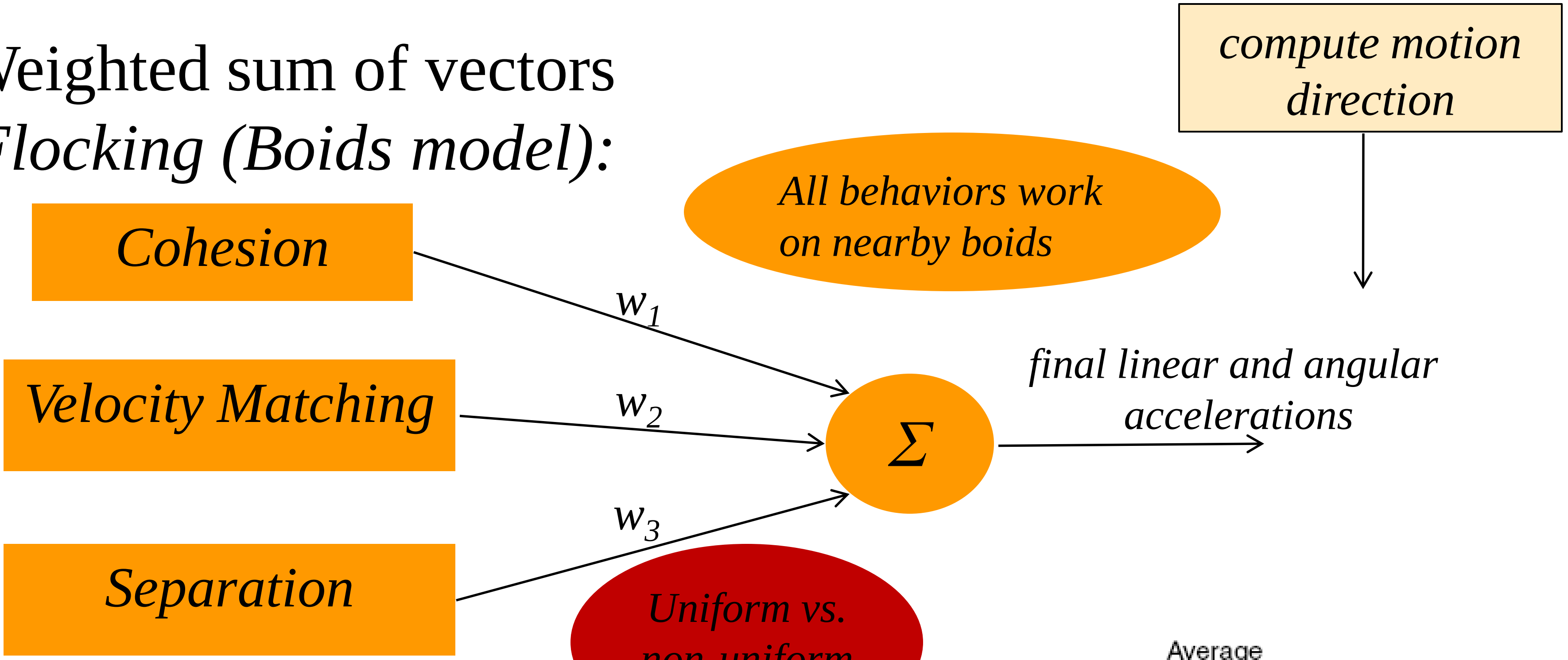


from "Artificial Intelligence for Games" by I. Millington & J. y

Combining Behaviors to Get Complex Behaviors

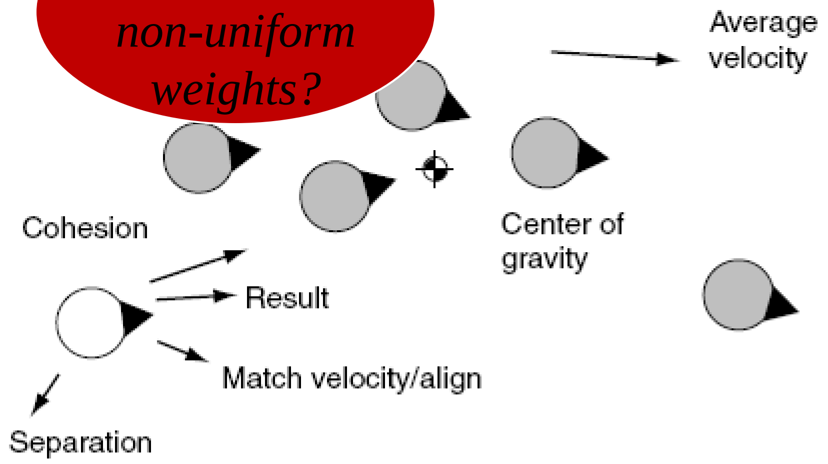
- Weighted sum of vectors

Flocking (Boids model):


compute motion direction
All behaviors work on nearby boids
Cohesion
Velocity Matching
Separation
w_1
w_2
w_3
Σ
final linear and angular accelerations

Uniform vs. non-uniform weights?

How to implement cohesion?

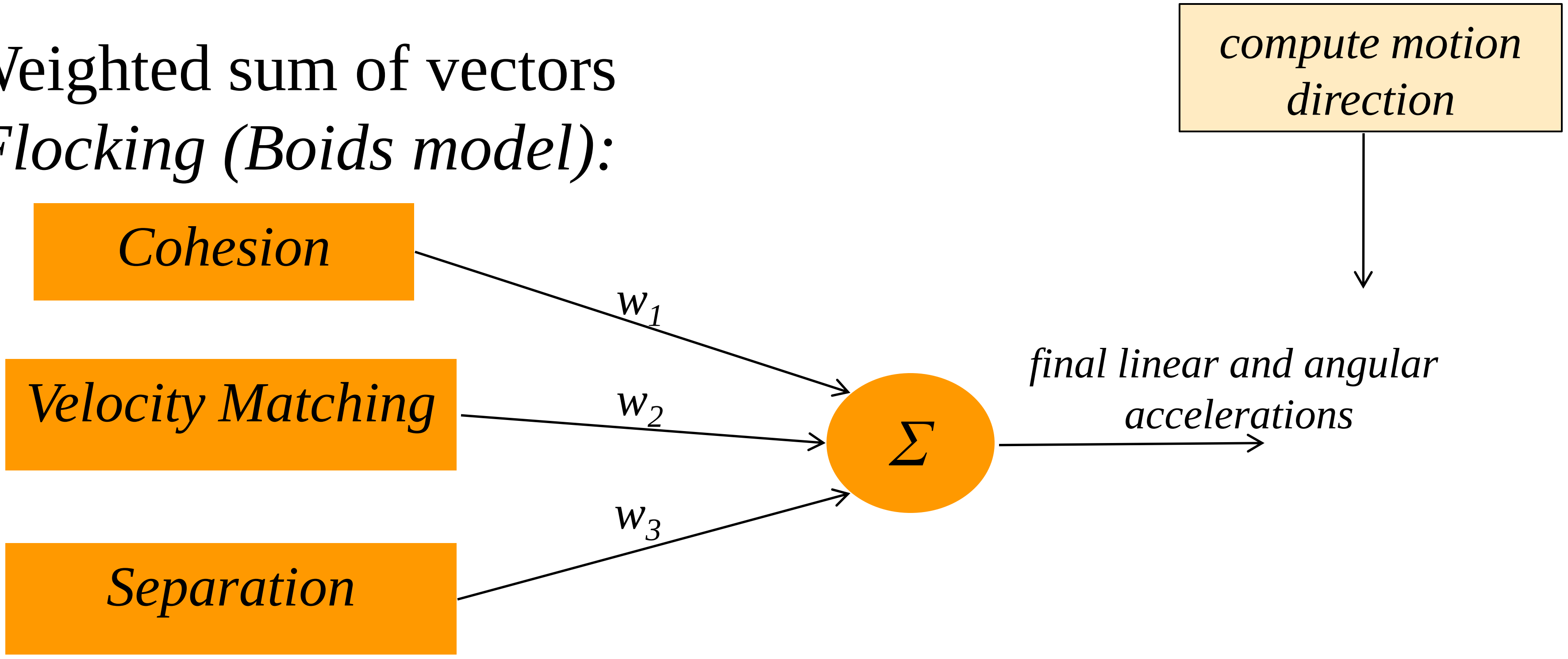


from "Artificial Intelligence for Games" by I. Millington & J. y

Combining Behaviors to Get Complex Behaviors

- Weighted sum of vectors

  Flocking (Boids model):



http://www.red3d.com/cwr/boids/

Combining Behaviors to Get Complex Behaviors

- Weighted sum of vectors

Any problems with weighted sum of behaviors?

compute motion direction

Combining Behaviors to Get Complex Behaviors

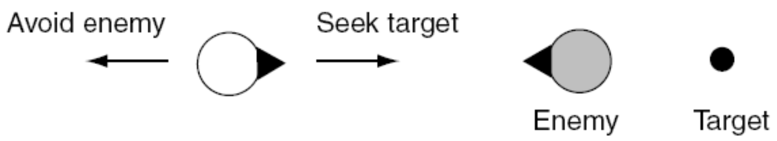
- Weighted sum of vectors

compute motion direction

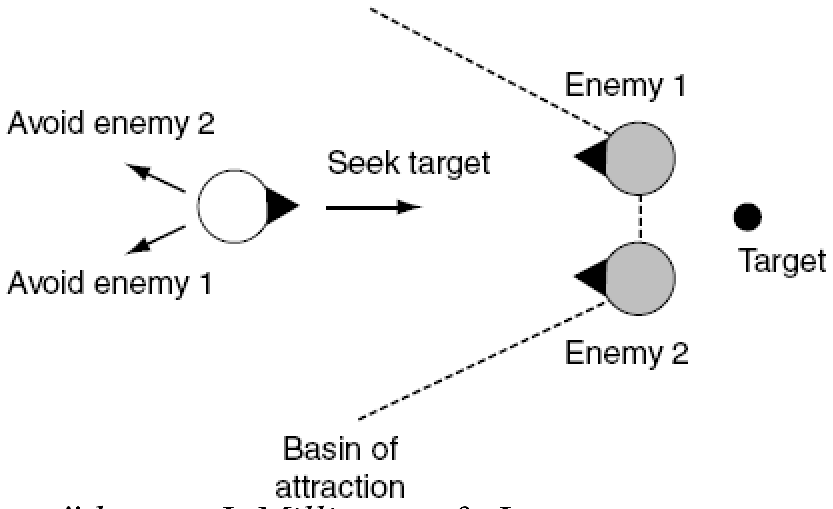
Any problems with weighted sum of behaviors?

Any solutions?

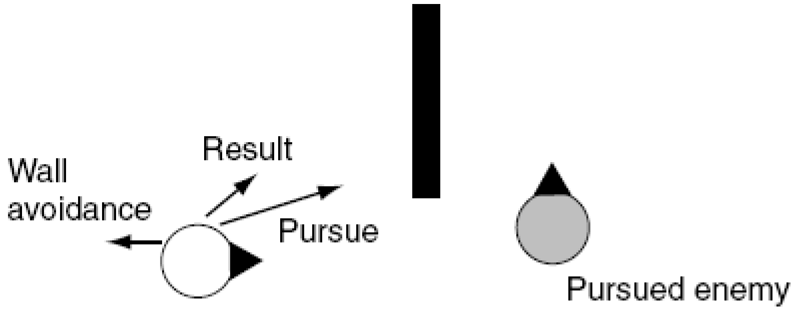
example 1:



example 3:



example 2:



from "Artificial Intelligence for Games" by I. Millington & J. y ge

Combining Behaviors to Get Complex Behaviors

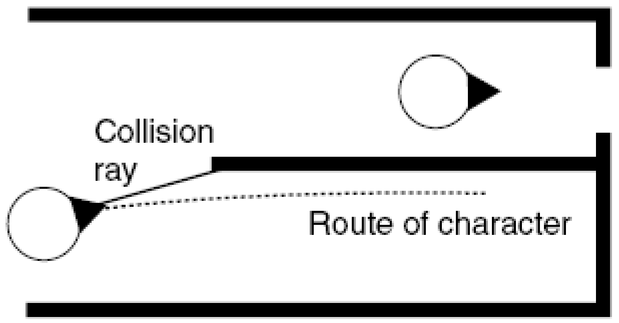
- Weighted sum of vectors

Any problems with weighted sum of behaviors?

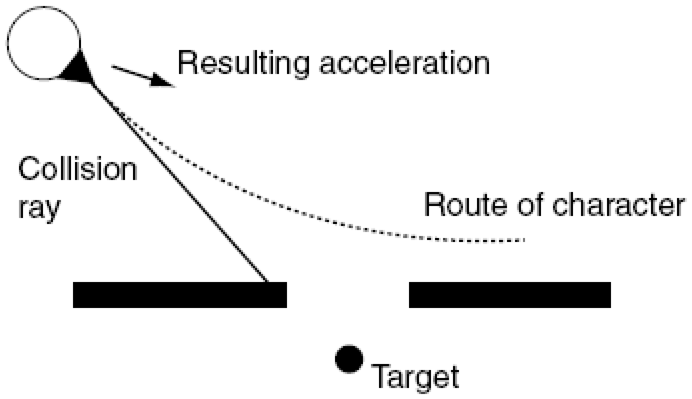
compute motion direction

Any solutions?

example 1:

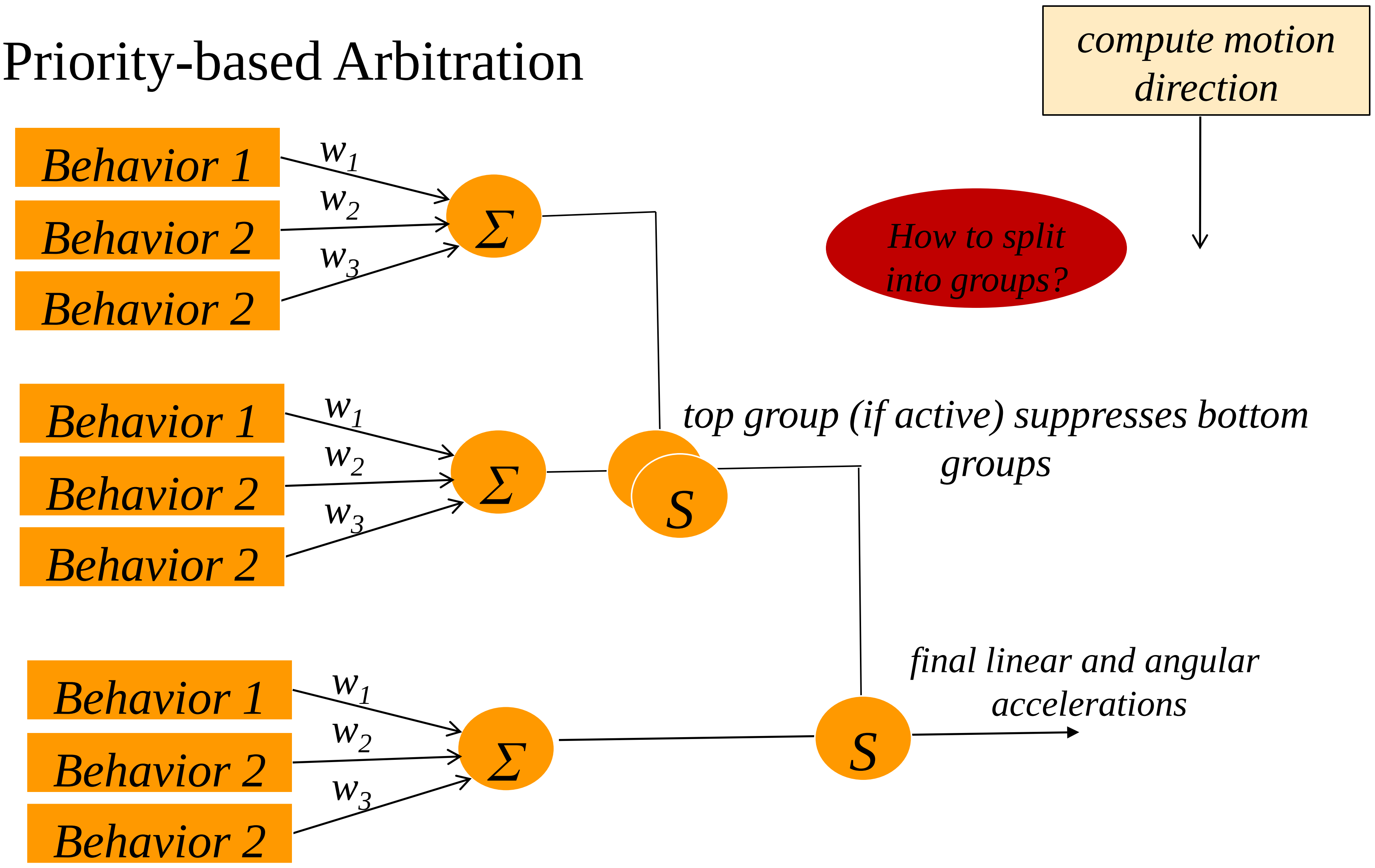


example 2:



Combining Behaviors to Get Complex Behaviors

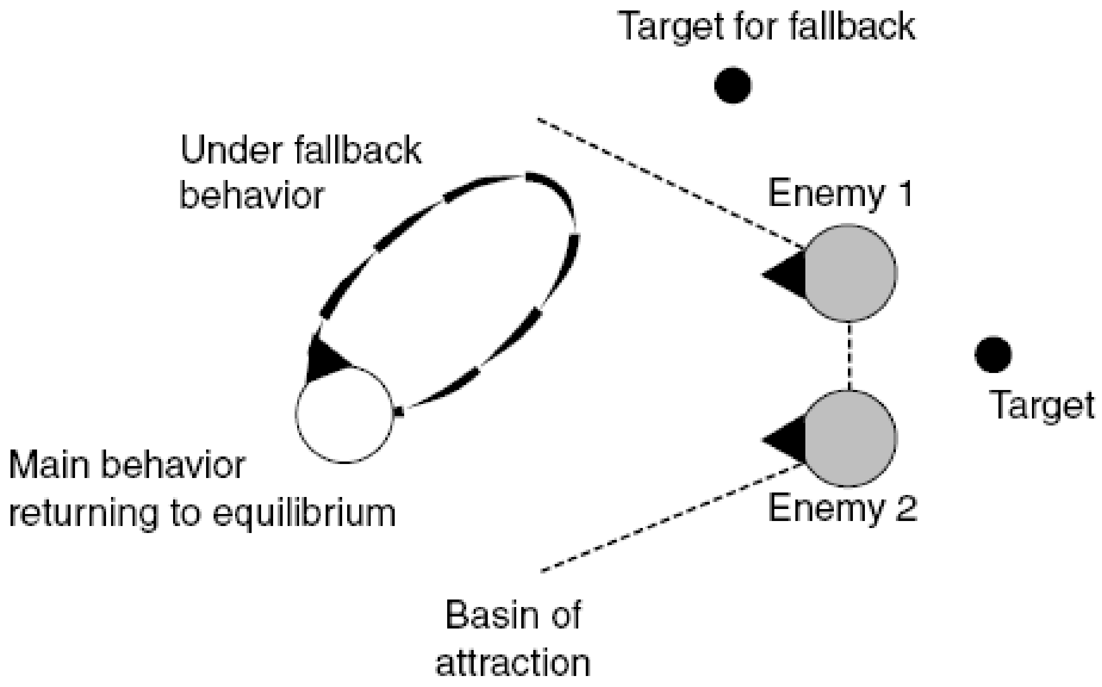
- Priority-based Arbitration



Combining Behaviors to Get Complex Behaviors

- Priority-based Arbitration

compute motion direction



from "Artificial Intelligence for Games" by I. Millington & J. y ge

Combining Behaviors to Get Complex Behaviors

- Examples of complex behaviors

compute motion direction

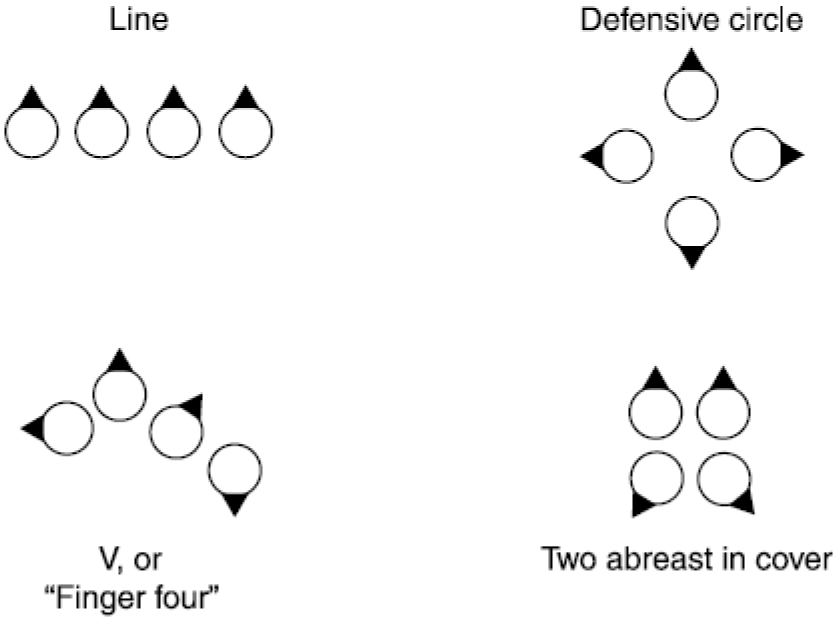
http://www.red3d.com/cwr/steer/

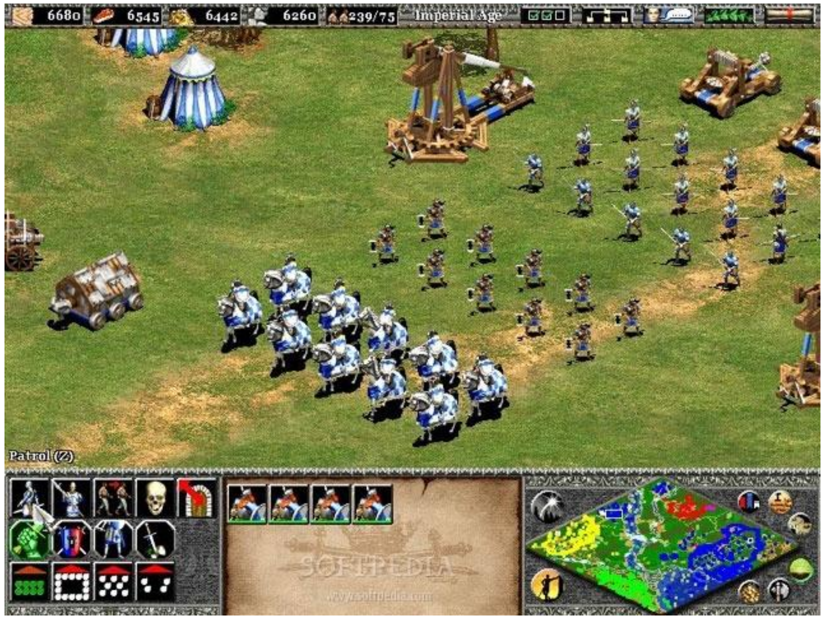
Formations

- Fixed Formation

compute motion direction

Any ideas how to do it?





from "Artificial Intelligence for Games" by I. Millington & J. y ge

Formations

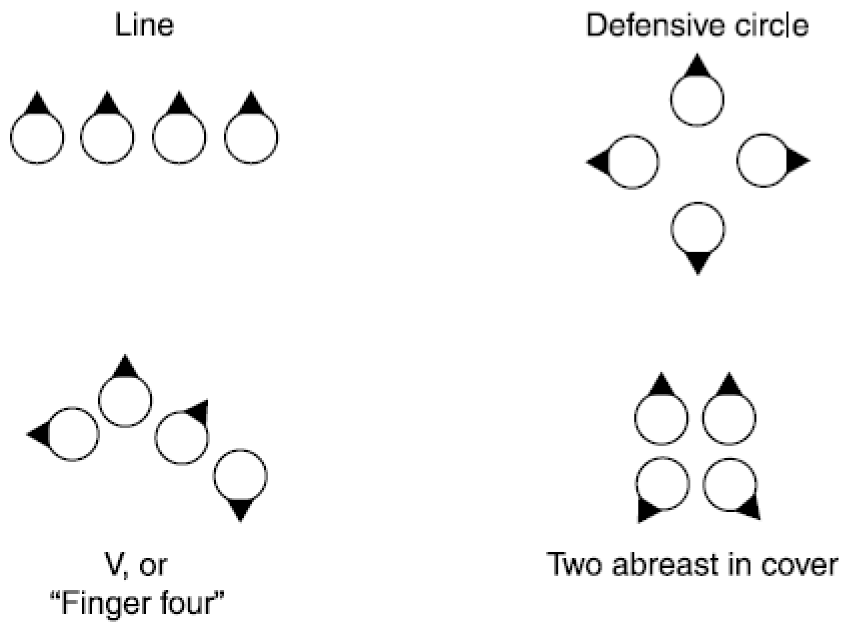
- Fixed Formation

  pick a leader

  define positions of others w.r.t. the leader position

Any problems?

compute motion direction



Formations

- Emergent Formation

Any problems?

compute motion direction

pick a leader

every other character selects the nearest (assigned) character and sets its own target w.r.t. it