15-466
Computer Game Programming

Movement: Behaviors, Crowds

Maxim Likhachev
Robotics Institute
Carnegie Mellon University
Dynamic Wander Behavior

• Move towards a random target

kinematic wander:

Unrealistic effects?

compute motion direction

from “Artificial Intelligence for Games” by I. Millington & J. Funge
Dynamic Wander Behavior

- Move towards a random target

kinematic wander:

full dynamic wander:

from “Artificial Intelligence for Games” by I. Millington & J. Funge
Dynamic Wander Behavior

• Move towards a random target
  \[ \lambda = \text{random with bias towards 0} \]
  \[ \text{target} = P + \overrightarrow{a} + \overrightarrow{b} \]
  \[ A = \text{max. acceleration} \times \text{normalize(target)} \]
  \[ dd\Psi = K\beta \text{ limited by max. angular acceleration} \]

full dynamic wander:

from “Artificial Intelligence for Games” by I. Millington & J. Funge
Dynamic Wander Behavior

• Move towards a random target
  \[ \lambda = \text{random with bias towards 0} \]
  \[ \text{target} = P + \vec{a} + \vec{b} \]
  \[ A = \max \text{acceleration} \ast \text{normalize(target)} \]
  \[ \dd \Psi = K \beta \text{ limited by max. angular acceleration} \]

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

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

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

compute motion direction

from “Artificial Intelligence for Games” by I. Millington & J. Funge
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. Funge
Path Following

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

\[ \text{compute nearest point } P_{\text{near}} \text{ on the path} \]

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

\[ \text{execute seek(target)} \]
Path Following

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

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

from “Artificial Intelligence for Games” by I. Millington & J. Funge
Path Following

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

compute nearest point $P_{\text{near}}$ on the path

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

execute seek(target)

from “Artificial Intelligence for Games” by I. Millington & J. Funge
Path Following

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

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

How to find a nearest point?
Any issues?
Any solutions?

\[ \text{when computing } P_{\text{near}} \text{ only search small path segment in front of previous } P_{\text{near}} \]

from “Artificial Intelligence for Games” by I. Millington & J. Funge
Path Following

• 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)

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

• Maintain distance from nearby characters
  for all nearby characters $T$
  
  strength = \( \min(K/d^2, \text{max. accel}) \)
  
  \( A = -\text{strength} \times \text{normalize}([dx,dz]) \)
  
  \( dd\Psi = K_1(\Psi - \lambda - \pi) \) limited by max. angular acceleration
Collision Avoidance using Separation

• Avoid collisions

  for all characters $T$ within cone of view
  run separation behavior

from “Artificial Intelligence for Games” by I. Millington & J. Funge
Collision Avoidance using Separation

- Avoid collisions for all characters $T$ within cone of view run separation behavior

from “Artificial Intelligence for Games” by I. Millington & J. Funge
Collision Avoidance with Collision Prediction

• Avoid collisions

\[
\text{for all characters with small } t_{\text{closest}}
\]

\[
B_{\text{closest}} = B_{\text{current}} + v_B^* t_{\text{closest}}
\]

\[
A_{\text{closest}} = A_{\text{current}} + v_A^* t_{\text{closest}}
\]

Flee as if character at \( A_{\text{closest}} \) and target at \( B_{\text{closest}} \)

\[
t_{\text{closest}} = -\frac{dp \cdot dv}{dv^2}, \text{ where}
\]

\[
dp = (B_{\text{current}} - A_{\text{current}})
\]

\[
dv = v_B - v_A
\]

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

What to do if many characters nearby?
Obstacle Avoidance

• Avoiding collisions with obstacles

compute motion direction

Any ideas how to do it?

Obs.

P
Obstacle Avoidance

• Avoiding collisions with obstacles

\[ \text{for all obstacles that can be approximated with a circle} \]
\[ \text{run separation behavior} \]
Obstacle Avoidance

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

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

• Avoiding collisions with obstacles

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

compute motion
direction

Failure
examples with
non-circular
obstacles?
Obstacle Avoidance

• Avoiding collisions with obstacles

   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

  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. Funge
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 $T$

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

from “Artificial Intelligence for Games” by I. Millington & J. Funge
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 $T$

from “Artificial Intelligence for Games” by I. Millington & J. Funge
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 $T$

http://www.red3d.com/cwr/steer/Wall.html
Combining Behaviors to Get Complex Behaviors

- Follow Path
- Avoid Obstacles
- Separation

...
Combining Behaviors to Get Complex Behaviors

• Weighted sum of vectors

- Follow Path
- Avoid Obstacles
- Separation

\[ \sum w_1 + w_2 + w_3 \]

- Any ideas how to pick weights?
- compute motion direction
- final linear and angular accelerations
Combining Behaviors to Get Complex Behaviors

- Weighted sum of vectors

**Flocking (Boids model):**

\[ \sum w_1 w_2 w_3 \]

- Cohesion
- Velocity Matching
- Separation

compute direction

final linear and angular accelerations

from “Artificial Intelligence for Games” by I. Millington & J. Funge
Combining Behaviors to Get Complex Behaviors

- Weighted sum of vectors
  
  **Flocking (Boids model):**

  \[
  \sum w_1 v_1 + w_2 v_2 + w_3 v_3
  \]

  - **Cohesion**: All behaviors work on nearby boids
  - **Velocity Matching**: Compute motion direction
  - **Separation**: Final linear and angular accelerations

  *How to implement cohesion?*

  *Uniform vs. non-uniform weights?*

  *from “Artificial Intelligence for Games” by I. Millington & J. Funge*
Combining Behaviors to Get Complex Behaviors

• Weighted sum of vectors

Flocking (Boids model):

- Cohesion
- Velocity Matching
- Separation

\[ \Sigma w_1 w_2 w_3 \]

compute motion direction

final linear and angular accelerations

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

example 1:
Avoid enemy → Seek target

Enemy, Target

example 2:
Wall avoidance → Result → Pursue

Pursued enemy

example 3:
Avoid enemy 2 → Seek target → Avoid enemy 1

Enemy 1, Enemy 2, Target

Any problems with weighted sum of behaviors?

Any solutions?

from “Artificial Intelligence for Games” by I. Millington & J. Funge
Combining Behaviors to Get Complex Behaviors

- Weighted sum of vectors

Any problems with weighted sum of behaviors?

Any solutions?

compute motion direction

deprecated

example 1:

example 2:

from “Artificial Intelligence for Games” by I. Millington & J. Funge
Combining Behaviors to Get Complex Behaviors

- Priority-based Arbitration

\[ \text{Behavior 1} \stackrel{w_1}{\rightarrow} \sum \quad \text{Behavior 2} \stackrel{w_2}{\rightarrow} \sum \quad \text{Behavior 2} \stackrel{w_3}{\rightarrow} \sum \]

\[ \text{Behavior 1} \stackrel{w_1}{\rightarrow} \sum \quad \text{Behavior 2} \stackrel{w_2}{\rightarrow} \sum \quad \text{Behavior 2} \stackrel{w_3}{\rightarrow} \sum \]

\[ \text{Behavior 1} \stackrel{w_1}{\rightarrow} \sum \quad \text{Behavior 2} \stackrel{w_2}{\rightarrow} \sum \quad \text{Behavior 2} \stackrel{w_3}{\rightarrow} \sum \]

How to split into groups?

top group (if active) suppresses bottom groups

compute motion direction

final linear and angular accelerations
Combining Behaviors to Get Complex Behaviors

- Priority-based Arbitration

*compute motion direction*

*from “Artificial Intelligence for Games” by I. Millington & J. Funge*
Combining Behaviors to Get Complex Behaviors

• Examples of complex behaviors

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

• Fixed Formation

Any ideas how to do it?

compute motion direction

from “Artificial Intelligence for Games” by I. Millington & J. Funge
Formations

- **Fixed Formation**
  
  *pick a leader*
  *define positions of others w.r.t. the leader position*

- Any problems?

- compute motion direction

---

*from “Artificial Intelligence for Games” by I. Millington & J. Funge*
Formations

• Emergent Formation

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

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