STEERING BEHAVIORS

Markéta Popelová, marketa.popelova [zavináč] matfyz.cz

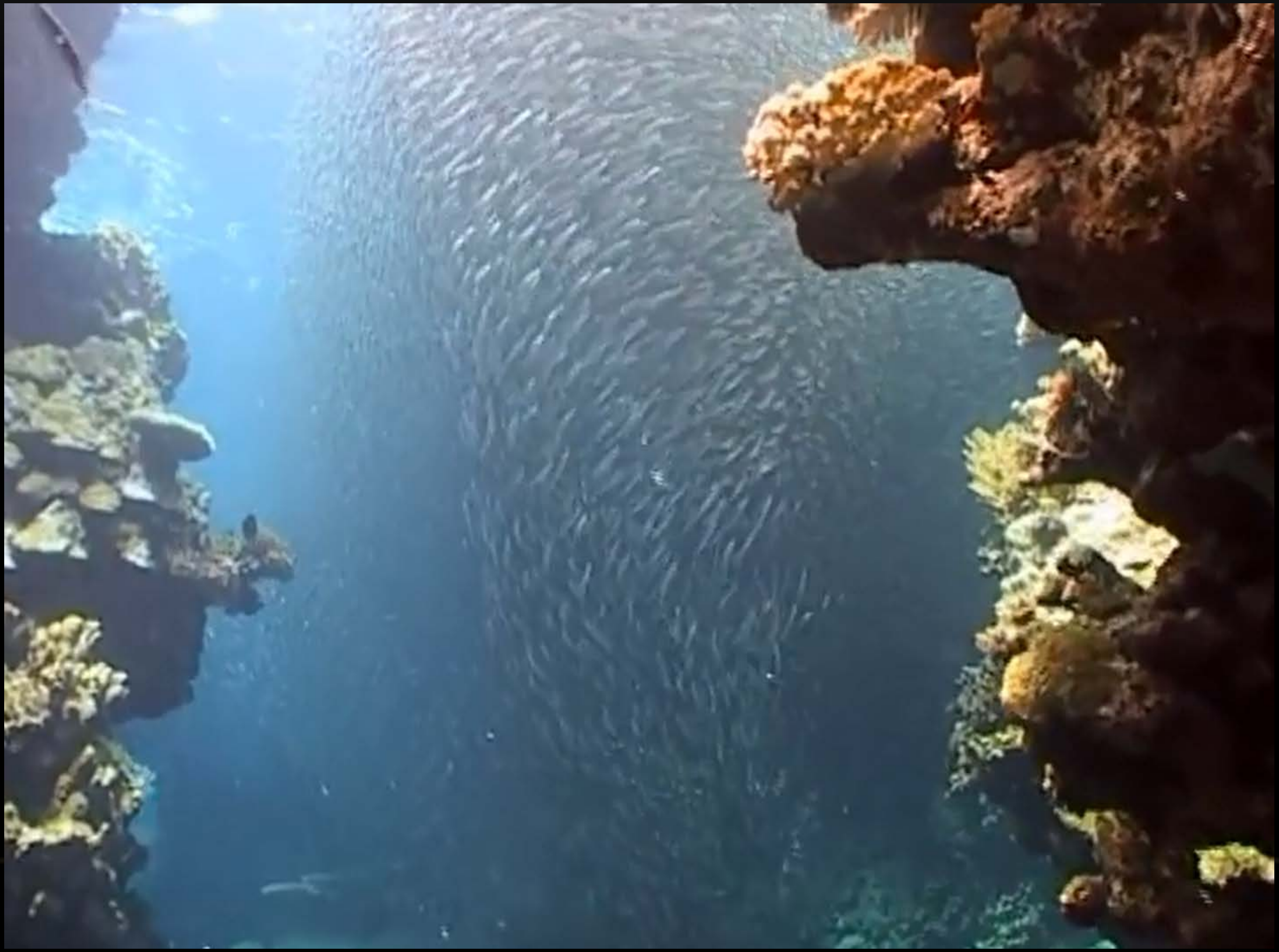
2013, Human-like Artificial Agents, MFF UK

Jakub Gemrot, gemrot@gamedev.cuni.cz

2015, Human-like Artificial Agents, MFF UK

Red Sea, Fish School
https://www.youtube.com/watch?v=PLkpfKiinTA

MOTIVATION

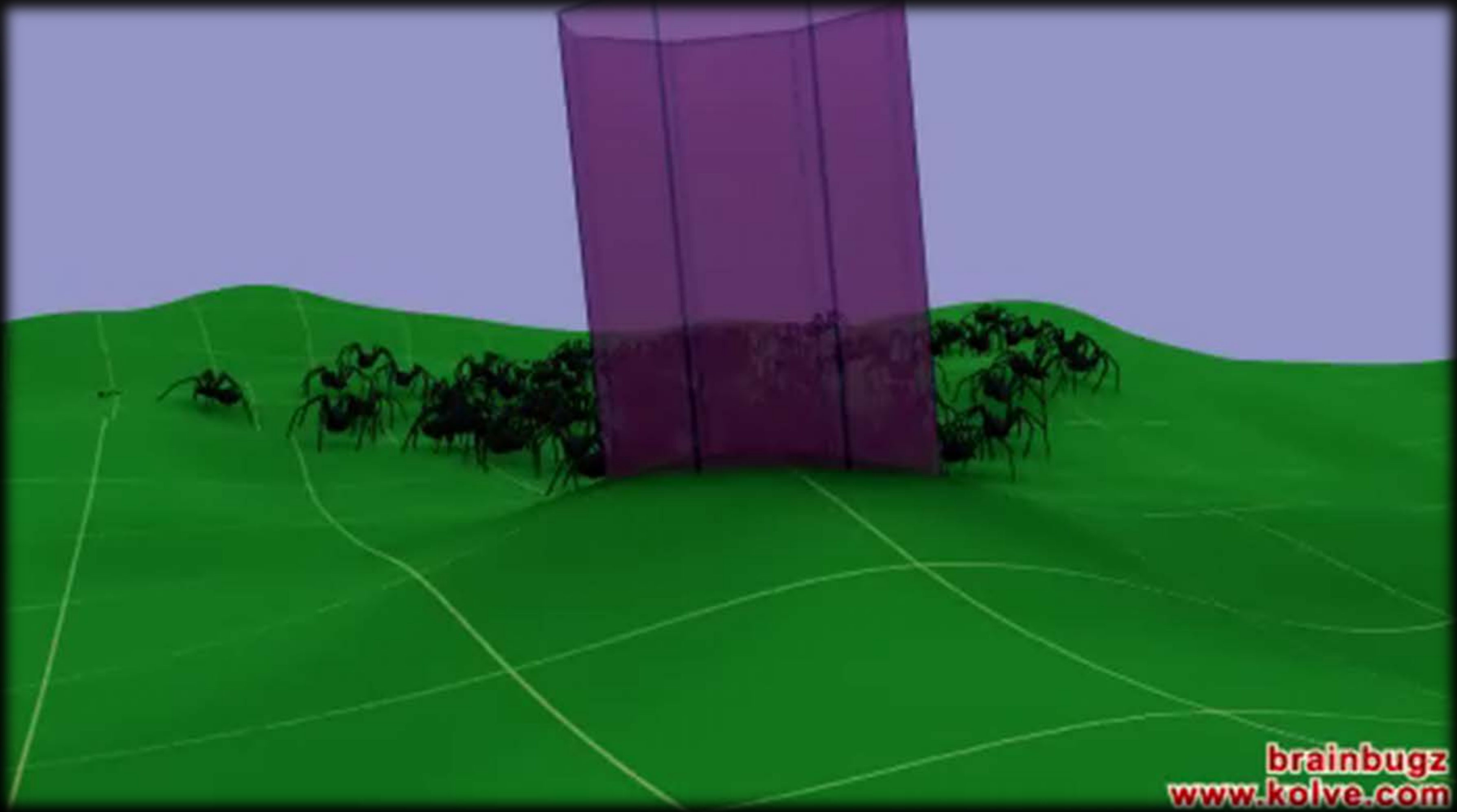


Boid Particles Simulation, Blender
https://www.youtube.com/watch?v=rLBrK0K-Ny8

MOTIVATION

BrainBugz
https://www.youtube.com/watch?v=uRTo9qECQrM

MOTIVATION

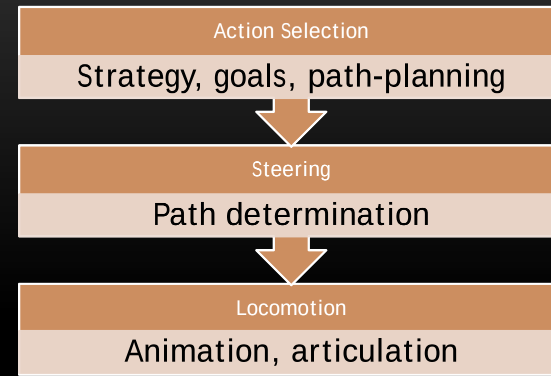


REQUIREMENTS FOR MOTION CONTROL

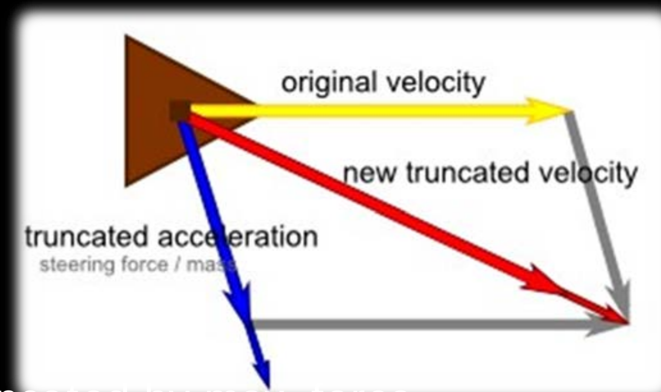
- Responding to dynamic environment
  - Avoiding obstacles and other agents
  - Interaction with environment and other agents
- Motion believability
- Speed of computation

- → One possible solution: Steering Behaviors by Craig W. Reynolds
  - 1986 Flocks, Herds, and Schools: A Distributed Behavioral Model [1]
    - Boids & Flocking Model
  - 1999 Steering Behaviors For Autonomous Characters [2]

STEERING BEHAVIORS - BASICS



- Hierarchy of motion behavior
  - Action selection layer + Path-finding
  - Steering (navigation) layer
  - Locomotion layer
- Simple vehicle model
  - Scalars: mass, max_force, max_speed
  - Vectors: location, velocity, orientation
- One steering force:
  - acceleration = steering_force / mass → truncated by max_force
  - new_velocity = original_velocity + acceleration → truncated by max_speed
  - new_location = original_location + new_velocity
  - *Or we can use forces as direct inputs for physical simulator*



BOIDS & FLOCKING MODEL

- Boid (bird like object)
- Flocking Model → 3 steering rules

Separation

- Do not get too close to nearby flockmates
- Steers boid from too close flockmates

Alignment

- Try to move at the same speed and direction (velocity) as nearby flockmates
- Steers boid to have the same velocity as the average of velocities of nearby flockmates

Cohesion

- Prefer to be at the center of the local flockmates
- Steers agent to the center of nearby flockmates

For figures and video see Craig Reynolds‘ web site http://www.red3d.com/cwr/boids/ [3]

Interactive Boids
http://blog.soulwire.co.uk/laboratory/flash/as3-flocking-steering-behaviors

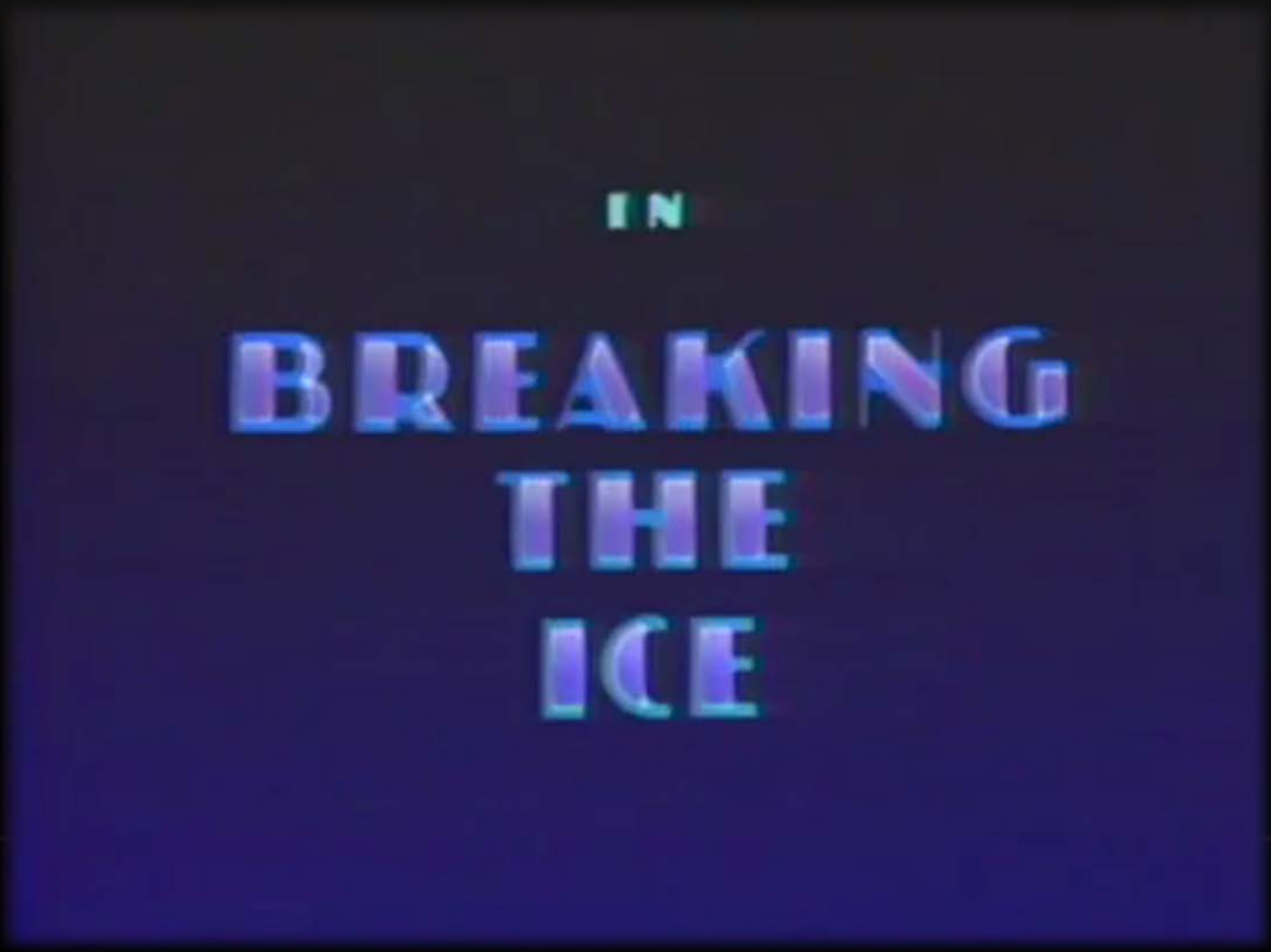
FLOCKING DEMONSTRATION I.

FLOCKING MODEL - FEATURES

- Relatively believable
- Relatively fast
  - Straightforward implementation → $O(n^2)$
  - Using spatial data structure for nearby flockmates detection → O(n)
- ➔ Used in films and games
  - E.g., Batman Returns

Stanley & Stella in: Breaking the Ice
http://www.youtube.com/watch?v=3bTqWsVqyzE

FLOCKING DEMONSTRATION II.



1999 C. REYNOLDS: STEERING BEHAVIORS FOR AUTONOMOUS AGENTS

- Seek & Flee
- Pursue & Evade
- Arrival
- Wander
- Obstacle Avoidance & Containment
- Collision Avoidance & Unaligned collision avoidance
- Wall Following
- Path Following
- Leader Following
- Flow Field Following

SEEK & FLEE

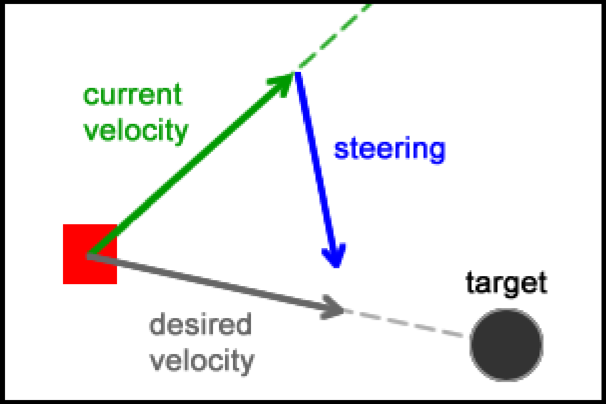
Seek steering force computation

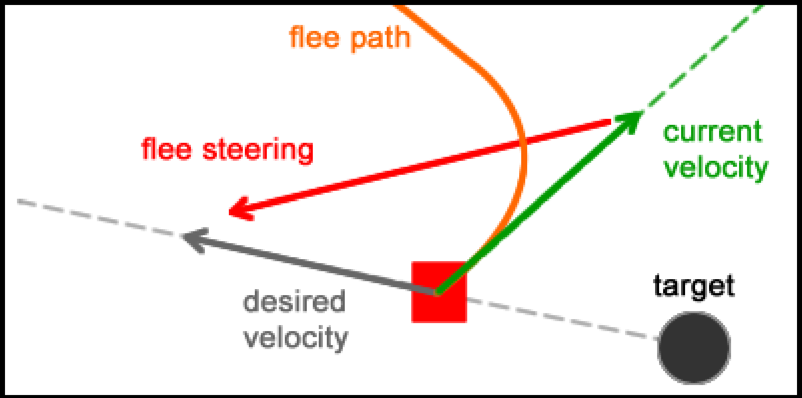
to_target = target_position – my_positin

desired_velocity = normalize(to_target) * max_speed

steering = desired_velocity – velocity

- Seek
  - steers agent to a static target
- Flee
  - steers agent from a static target





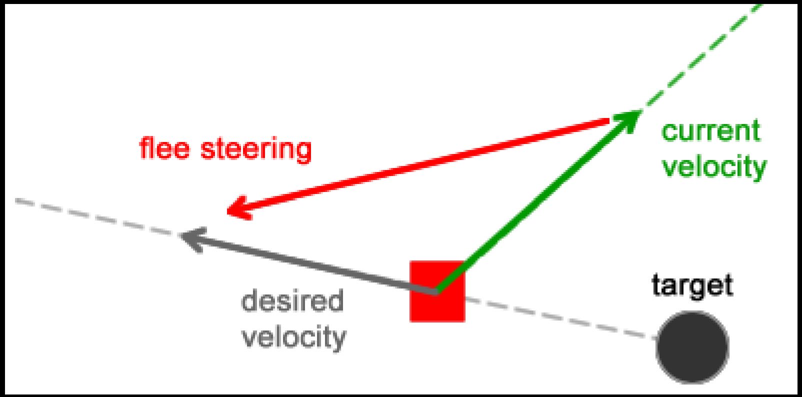
For figures and video see Craig Reynolds‘ web site
http://www.red3d.com/cwr/steer/SeekFlee.html [3]

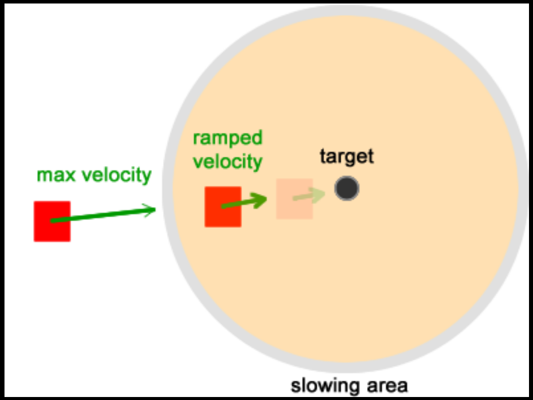
ARRIVAL

Arrival steering force computation

to_target = target_position – my_position
distance = length(to_target)
ramped_speed = max_speed * (distance / slowing_distance)
clipped_speed = min(ramped_speed, max_speed)
desired_velocity = to_target * (clipped_speed / distance)
steering_force = desired_velocity – velocity

- As Seek, except the agent slows down as it approaches the target

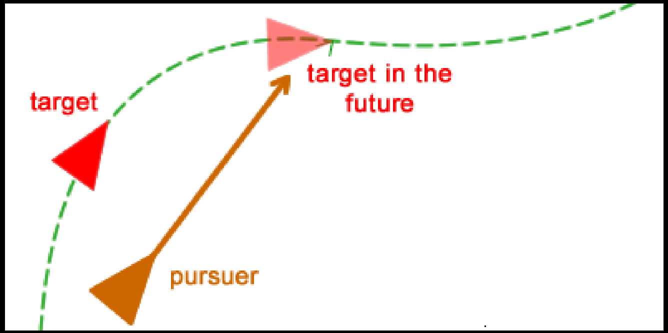


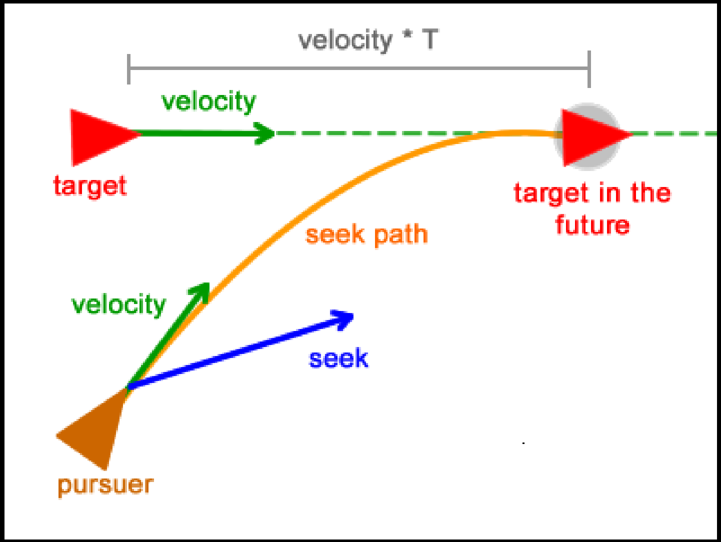


For figures and video see Craig Reynolds' web site
http://www.red3d.com/cwr/steer/Arrival.html [3]

PURSUE & EVADE

- As seek & flee, except the target moves
- Agent predicts the location of the target in the next tick of the simulation
- Prediction based on distance
  - Nearer the target is,
    less the prediction (T) is used





For figures and video see Craig Reynolds' web site
http://www.red3d.com/cwr/steer/PursueEvade.html [3]

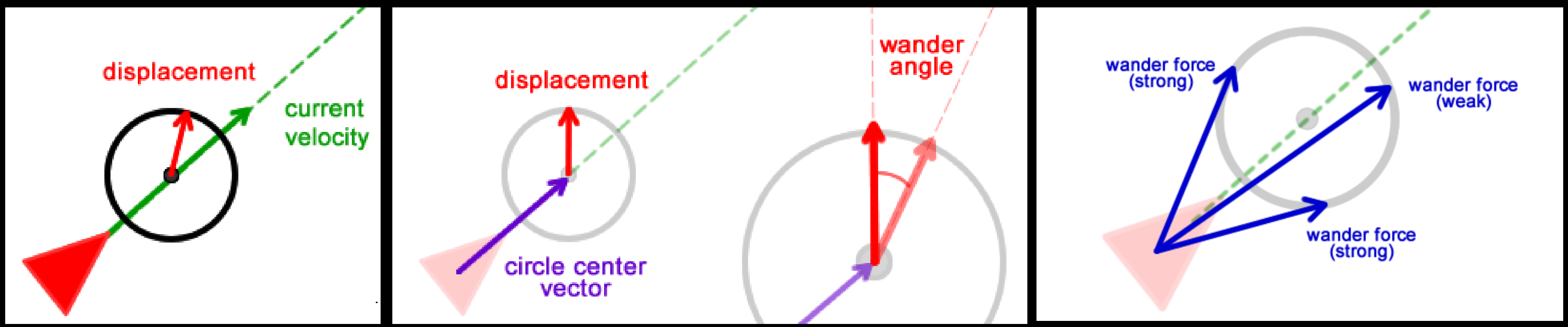
WANDER

- Type of random steering: the steering direction on one frame is related to the steering direction on the next frame
- More believable than totally random steering forces
- Steering force:
  - At each time step a random offset is added to the wander direction
  - The modified wander direction is constrained to lie on the big circle
- Constriction of the steering: big circle
- Constriction of the offset: small circle

For figures and video see Craig Reynolds‘ web site
http://www.red3d.com/cwr/steer/Wander.html [3]

WANDER

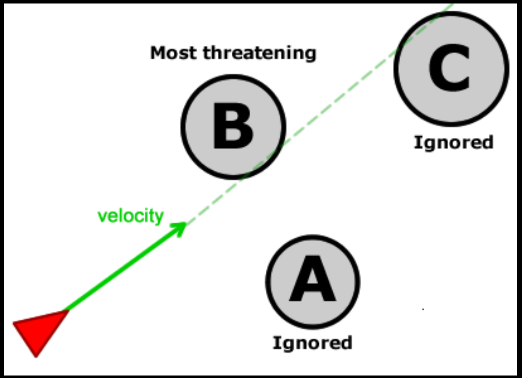
- Steering force:
  - At each time step a random offset is added to the wander direction
  - The modified wander direction is constrained to lie on the big circle



For figures and video see Craig Reynolds‘ web site
http://www.red3d.com/cwr/steer/Wander.html [3]

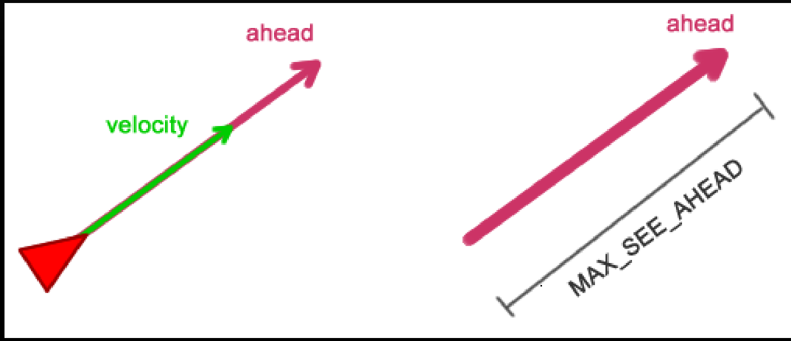
UNALIGNED COLLISION AVOIDANCE

- Separation
  - Agent is steered from too close neighbors
- Unaligned collision avoidance
  - Potential collisions with other agents are predicted
  - Agent is steered to avoid the site of the predicted collision

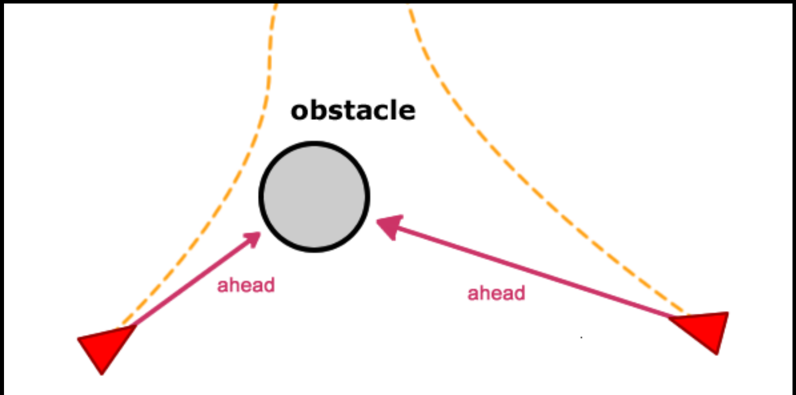


For figures and video see Craig Reynolds' web site
http://www.red3d.com/cwr/steer/Unaligned.html [3]

UNALIGNED COLLISION AVOIDANCE



- Seeing head
  - Defined by constant

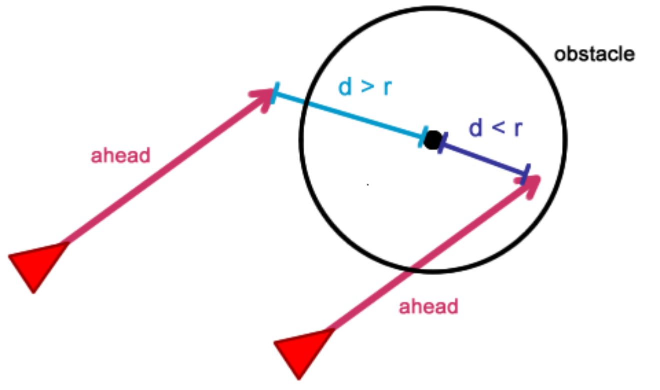


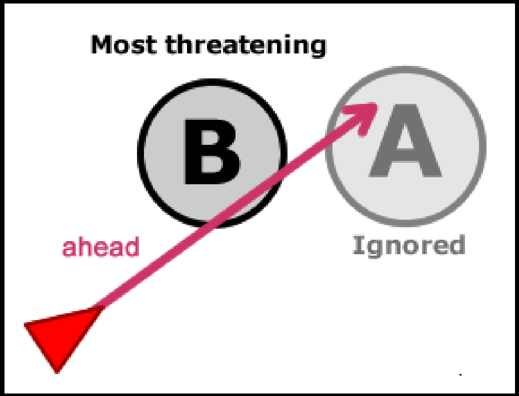
- Larger the constant
  - Sooner an obstacle is detected

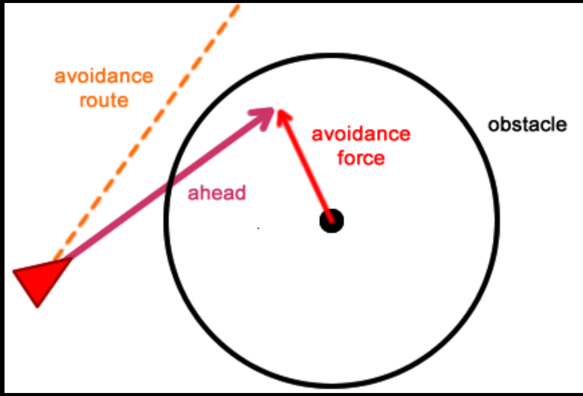
For figures and video see Craig Reynolds' web site http://www.red3d.com/cwr/steer/Unaligned.html [3]

UNALIGNED COLLISION AVOIDANCE

- Detection by simple relation of the distance to an obstacle center
- Only closest thread is considered
- And the avoidance force generated

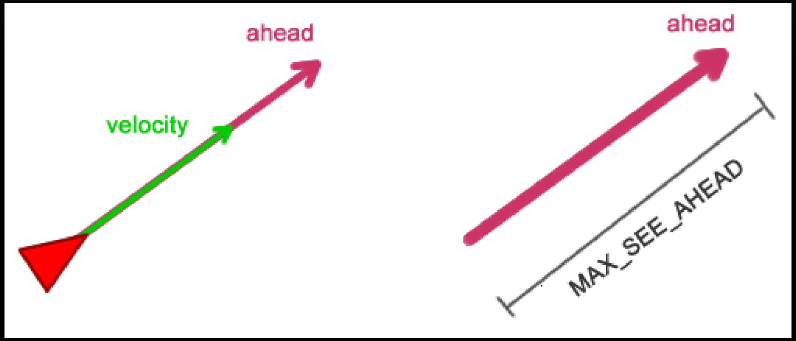




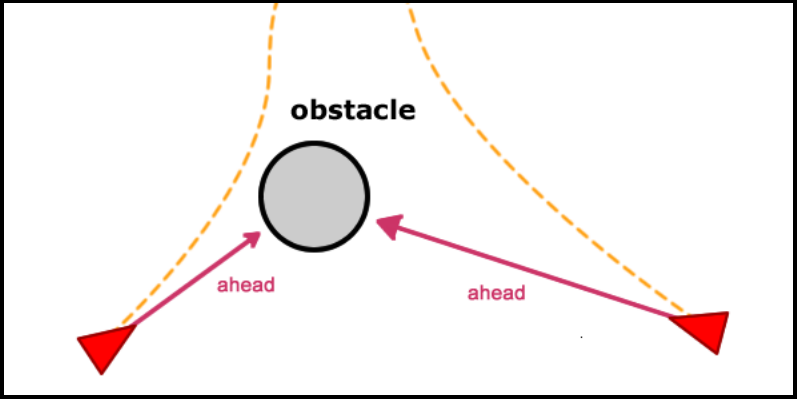


For figures and video see Craig Reynolds' web site
http://www.red3d.com/cwr/steer/Unaligned.html [3]

UNALIGNED COLLISION AVOIDANCE



- Seeing head
  - Defined by constant

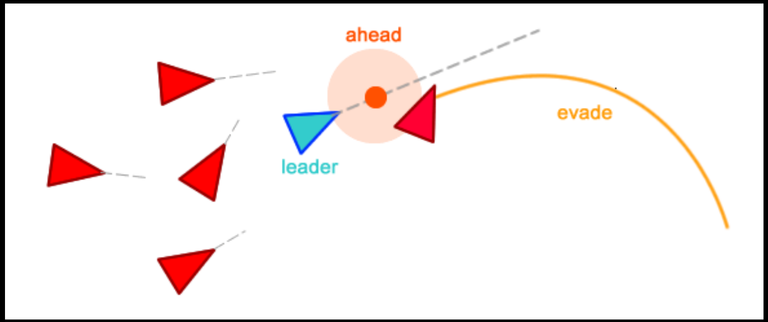


- Larger the constant
  - Sooner an obstacle is detected

For figures and video see Craig Reynolds' web site
http://www.red3d.com/cwr/steer/Unaligned.html [3]

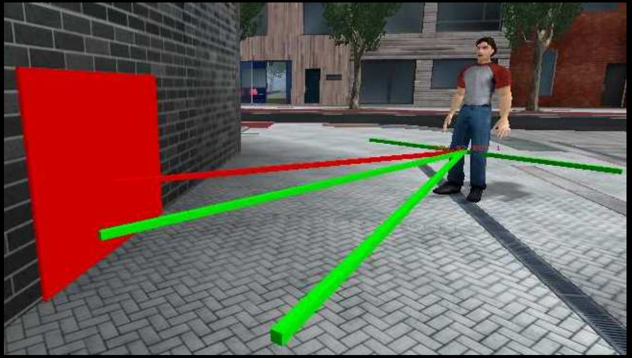
LEADER FOLLOWING

- Agent is steered to follow a Leader (grey).
- Steering force consists of:
  - Arrival – the target is slightly behind leader
  - Separation – to prevent collisions with other followers
  - If a follower finds itself in a rectangular region in front of the leader, it will steer laterally away from the leader's path



For figures and video see Craig Reynolds' web site
http://www.red3d.com/cwr/steer/LeaderFollow.html [3]

OBSTACLE AVOIDANCE



- Obstacle detection
  - Navigation graph, navigation mesh, etc.
  - Point content
  - Line traces
  - …
- Obstacle Avoidance by C. Reynolds
  - An imaginary cylinder in front of the agent should be free
  - If it is free, the steering force is zero vector
  - Otherwise it is the vector from the most threatening obstacle

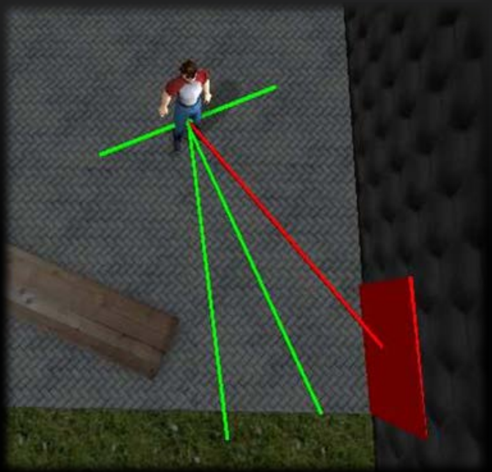
OBSTACLE AVOIDANCE & CONTAINMENT

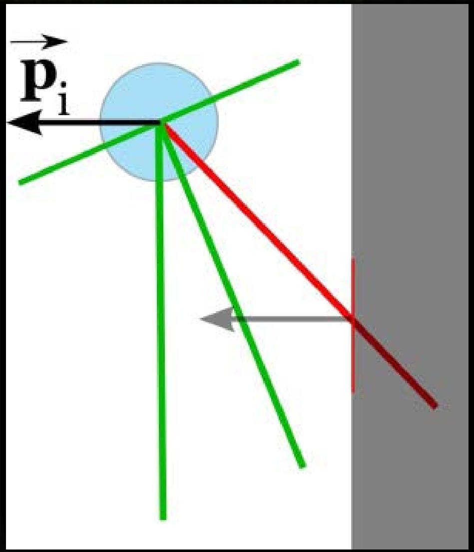
- The most threatening obstacle is detected and the agent is steered from it
- The agent's future position is predicted and the agent is steered towards the allowed region

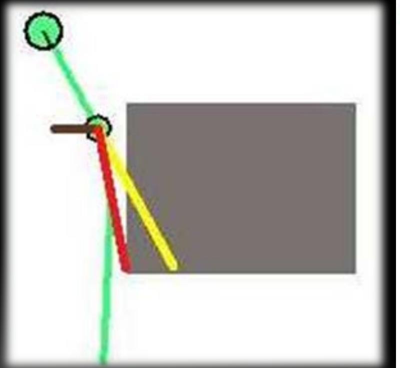
For figures and video see Craig Reynolds' web site
http://www.red3d.com/cwr/steer/Obstacle.html [3]
http://www.red3d.com/cwr/steer/Containment.html [3]

OBSTACLE AVOIDANCE

- Obstacle detection – typically fixed rays
- Possible implementation:
  - Steering force = $\sum_{i \in I} \left(\vec{p_i} \cdot W_i \cdot F \cdot \left(\frac{2 \cdot D_i}{R_i} \right)^{O} \right)$
  - I = *set of colliding rays*
  - W_i = *weight of the ray (front – bigger, side – lower)*
  - D_i = *length of the colliding ray part*
  - R_i = *ray length*
  - $\vec{p_i}$ = *normal of the obstacle*
  - O = *force order*
  - F = *basic magnitude of the force*





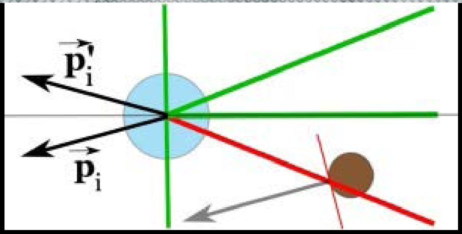


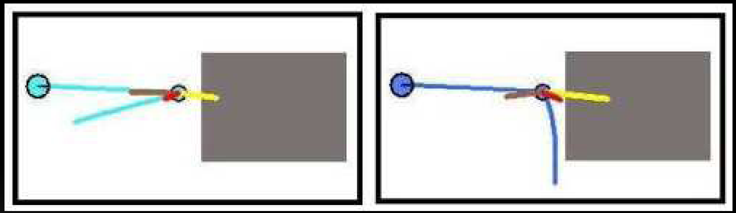
Hitman Blood Money, 2009
http://www.youtube.com/watch?v=ycDi7fK797U

(DYNAMIC) OBSTACLE AVOIDANCE EXAMPLE

PROBLEMS AND DISCUSSION OF OA USE

- Problems with obstacles detection
  - Narrow obstacles
    - Obstacles may not be detected
    - Obstacles may be detected wrongly
  - Ray length
    - Quick reactions vs. narrow passages
  - Simulation frequency
- Specific situations
  - Front collisions
- Local traps and complicated situations
  - OA uses only local information

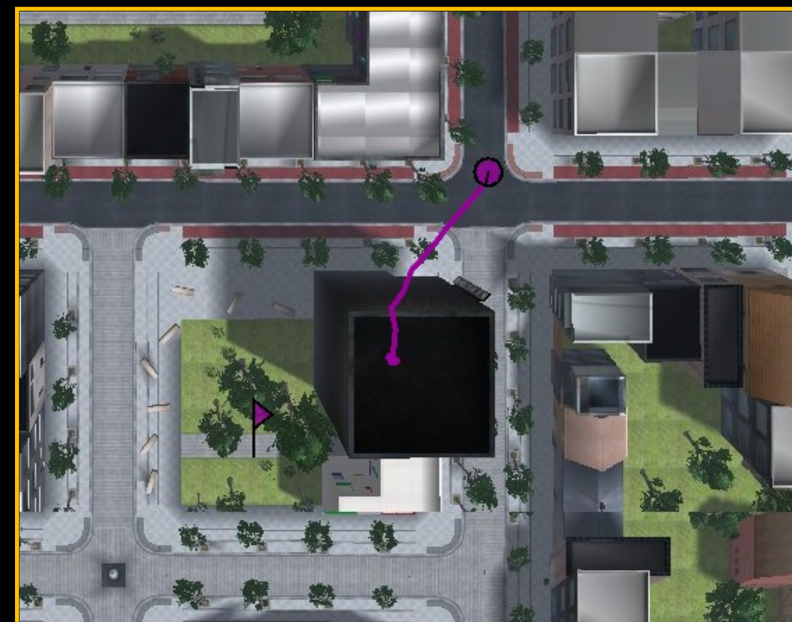




OBSTACLE AVOIDANCE & LOCAL TRAPS

OBSTACLE AVOIDANCE & LOCAL TRAPS

OBSTACLE AVOIDANCE & LOCAL TRAPS

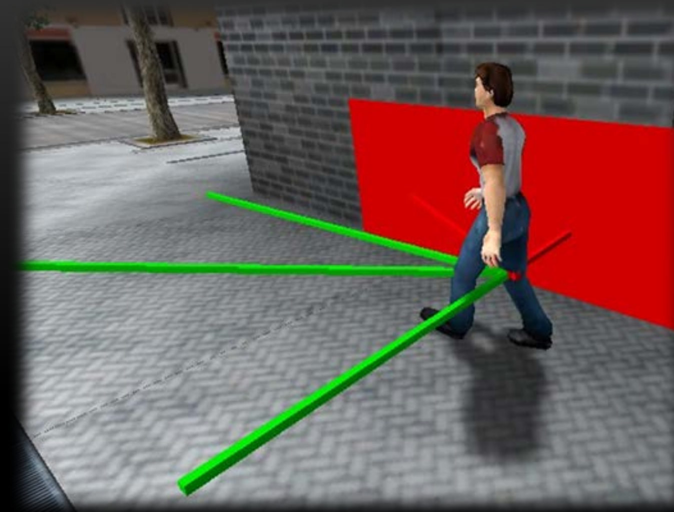


WALL FOLLOWING

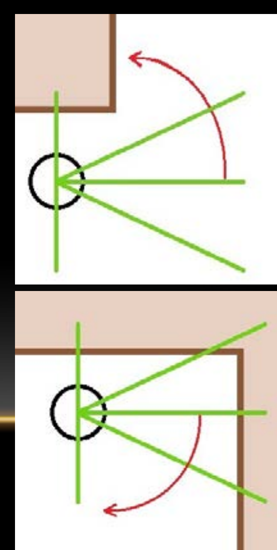
- Agent is steered to move in parallel with a wall
- The future agent's position is predicted (the black dot)
- This future position is projected to the nearest point on a wall (red dot)
- Red line represents the wall's normal and leads to the target point (red circle)
- Seek behavior is used to steer agent towards the target point
- Surface protocol:
  - the nearest point on the wall
  - the normal at that point

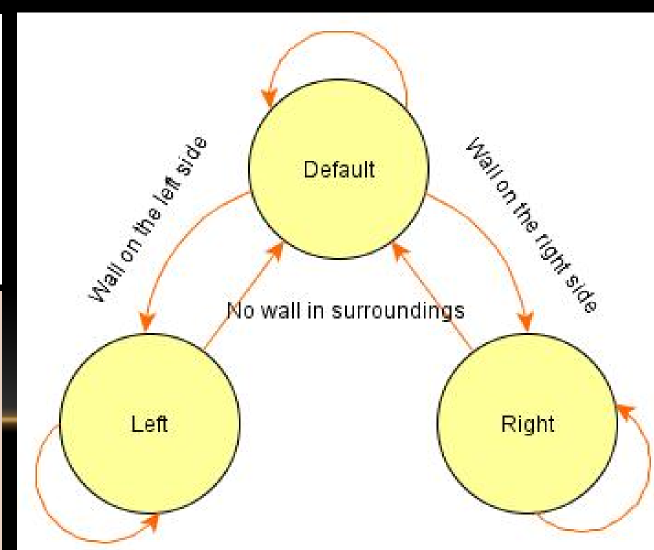
For figures and video see Craig Reynolds' web site
http://www.red3d.com/cwr/steer/Wall.html [3]

WALL FOLLOWING

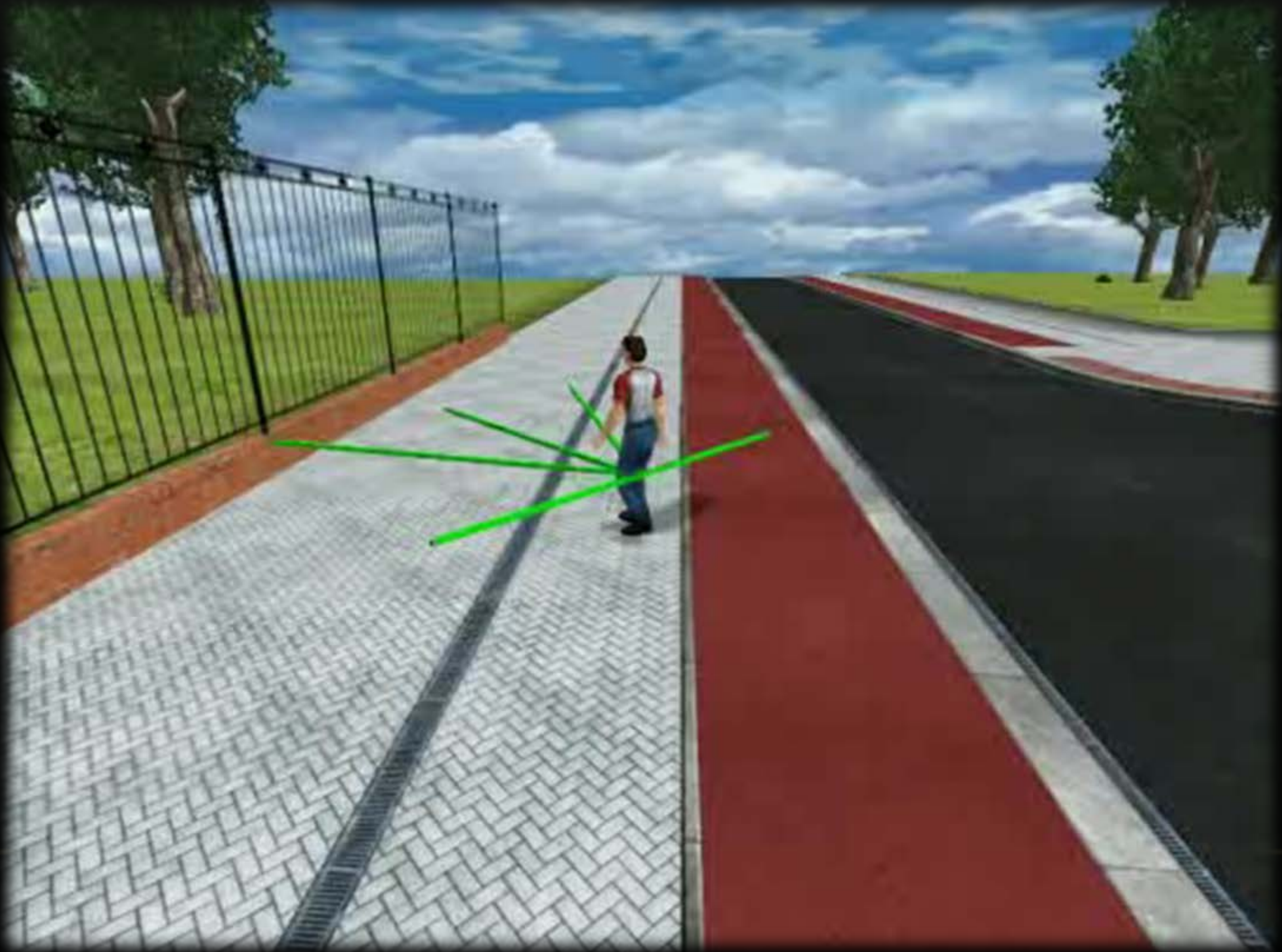


- Notes on possible implementation:
  - Wall is detected by rays
  - Attractive force to wall
    - The farther from wall an agent is, the bigger the attractive force is
  - Repulsive force from wall – if the agent is too close to wall
    - The closer to wall the agent is, the bigger is the repulsive force
  - Special situations
    - Edges
    - Front collisions

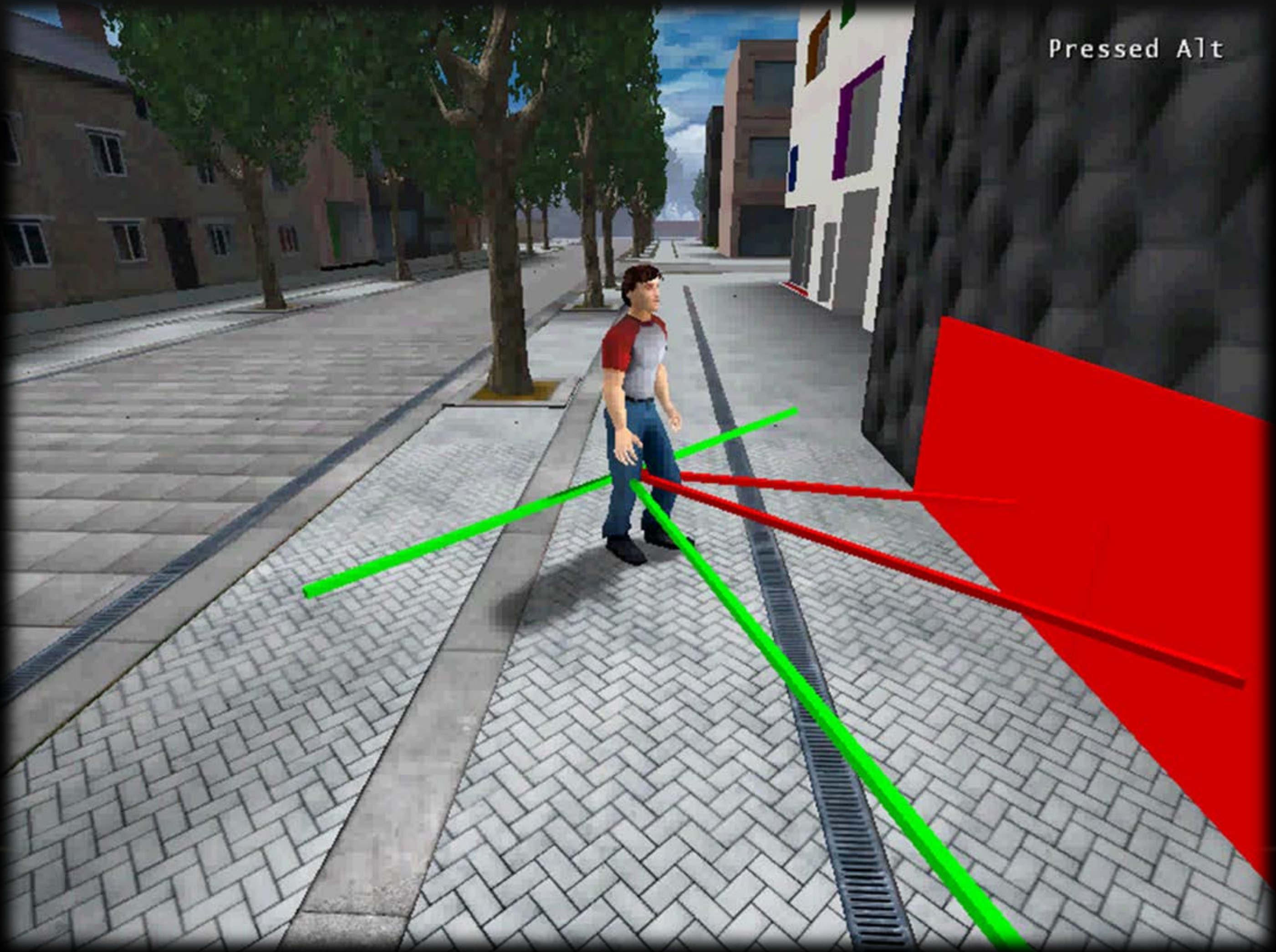




WALL FOLLOWING – DEMO I.



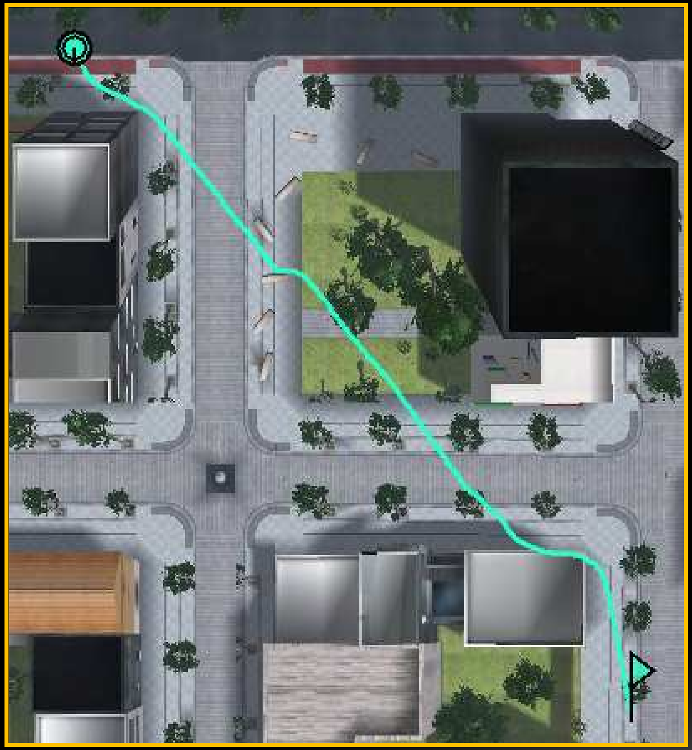
WALL FOLLOWING DEMO II.



WALL FOLLOWING IN COMBINATION

TA + OA

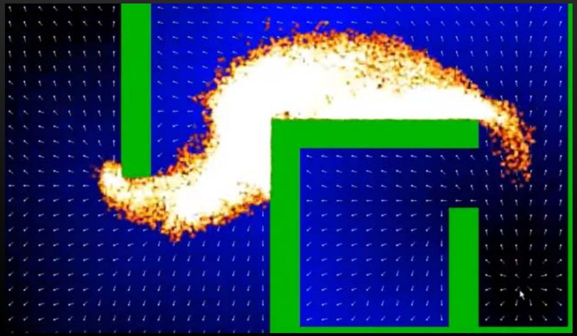
➔ directly through city



TA + WF

➔ on the sidewalks

FLOW FIELD FOLLOWING



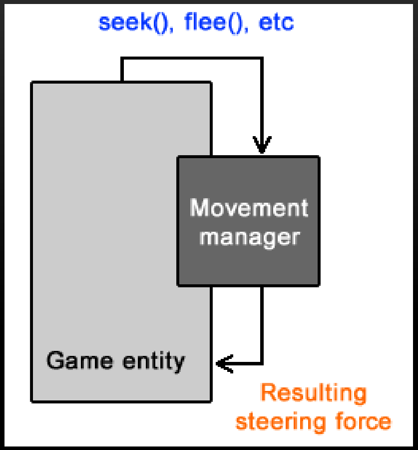
- Flow field defines mapping: location → flow vector
  - May be defined procedurally / based on data
  - May be static / time-varying
- The future location is predicted
- F = flow vector at this location
- steering_force = velocity – F

For figures and video see Craig Reynolds' web site
http://www.red3d.com/cwr/steer/FlowFollow.html [3]

Vector-field Pathfinding
https://www.youtube.com/watch?v=Bspb9g9nTto

Supreme Commander 2, 2010
http://www.youtube.com/watch?v=jA2epda-RkM

COMBINING STEERING BEHAVIORS



- Each steering behavior returns single vector (steering force)
- What to do with more steering behaviors?
  - Select and apply the most important steering behavior
  - Select random active steering behavior
  - Sum all forces together
    - → Average of all forces
    - → Average of all non-zero forces

- $v_t = \frac{w_0 \cdot v_{t-1} + \sum_{i \in I}(w_i \cdot s_i)}{w_0 + \sum_{i \in I} w_i}$
- $I = \{i = 1, \dots, n \mid s_i \neq 0\}$

- v_t = velocity in time t
- $s_1, \dots, s_n$ = steering forces
- w_i = weight of steering force s_i
- w_0 = weight of original velocity
- I = set of non-zero steering forces

STEERING BEHAVIORS FOR IVA‘S

- Which motion problems do we deal with in applications with IVA‘s?
- Where would be steering behaviors helpful?

REAL EXAMPLE I.

No collision avoidance
Stronghold Crusader Extreme, 2008
http://www.youtube.com/watch?v=IZpgMnu_lAk

REAL EXAMPLE II.

Small collision radius
Dawn of War, 2009
http://www.youtube.com/watch?v=IZpgMnu_lAk

REAL EXAMPLE III.

Primitive (and slow) collision avoidance
Knights and Merchants, 1998
http://www.youtube.com/watch?v=IZpgMnu_lAk

REAL EXAMPLE IV.

Getting stuck
Empire Total War, 2009
http://www.youtube.com/watch?v=IZpgMnu_lAk

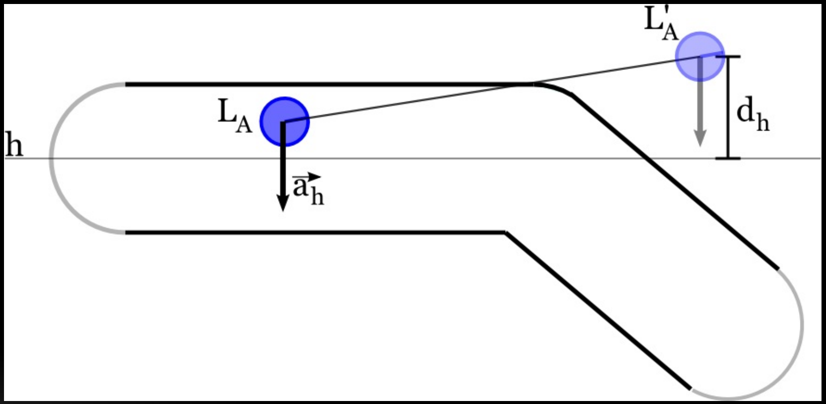
PROBLEMS OF LOCAL INFORMATION

- Complicated tasks can not be solved
- What to do?
  - → use global knowledge of the environment
  - → plan the path

DIRECT FOLLOWING OF THE PLANNED PATH

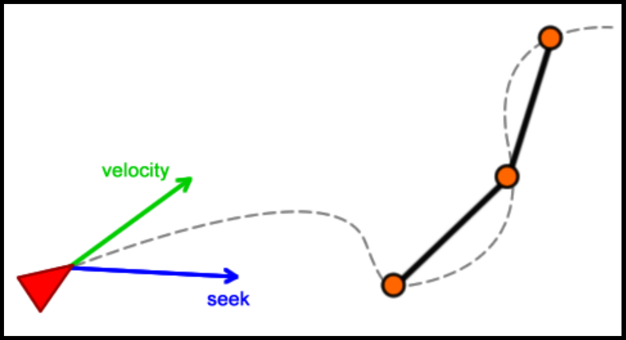
PROBLEMS OF DIRECT FOLLOWING

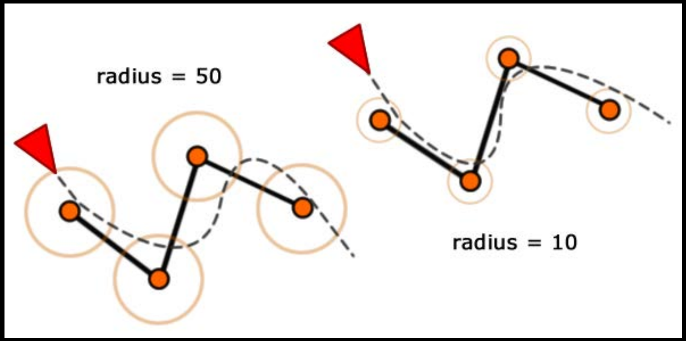
- Not believable
- Sometimes lacks smoothness
- What to do?
  - → steering behavior Path Following
    - Parameters: path (a list of locations), distance from path
    - Notes on implementation
      - Pair of path nodes
      - Force to the center axis
    - Improvements
      - Projection length
      - Regulation force



PATH FOLLOWING

- Agent is steered to move along the path in the given direction while keeping its center in the gray region





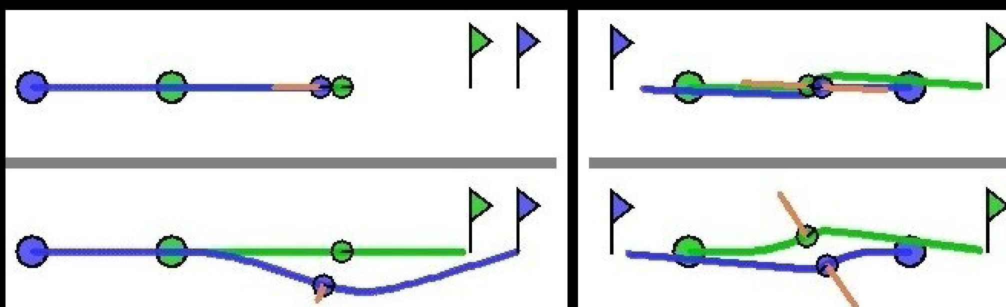
For figures and video see Craig Reynolds' web site
http://www.red3d.com/cwr/steer/PathFollow.html [3]
http://www.red3d.com/cwr/steer/CrowdPath.html [3]

GameDev Tutorials
http://gamedevelopment.tutsplus.com/tutorials/understanding-steering-behaviors-path-following--gamedev-8769

PEOPLE (COLLISION) AVOIDANCE

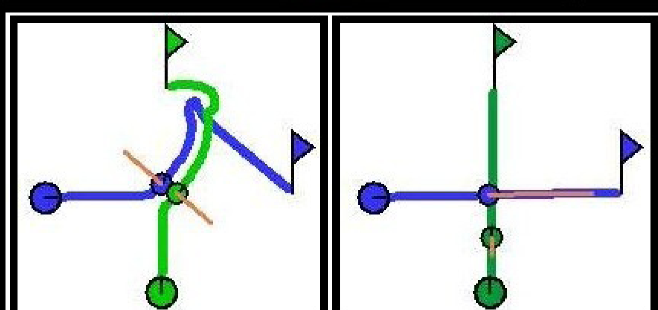
- Basics
  - Repulsive force from other too close agents
- Problems

1

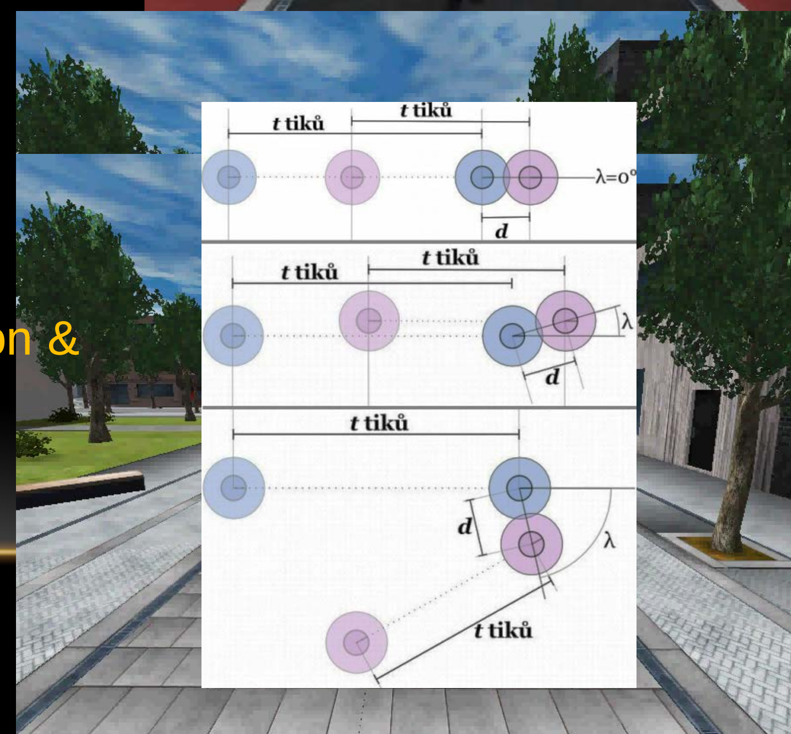


→ Circumvention (rotational force)

2



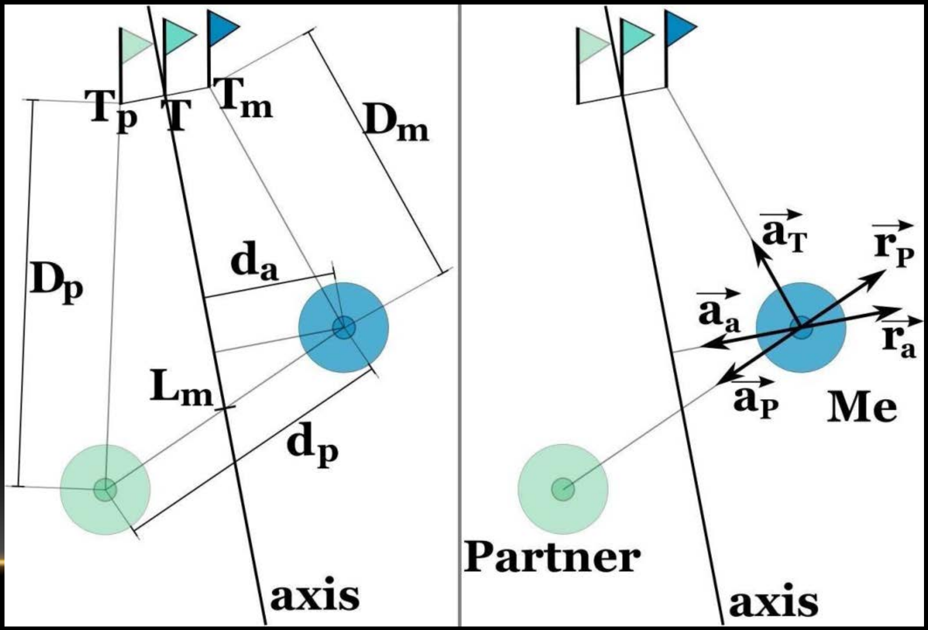
→ Acceleration & deceleration



PEOPLE AVOIDANCE – DEMONSTRATION

OTHER SOCIAL INTERACTIONS

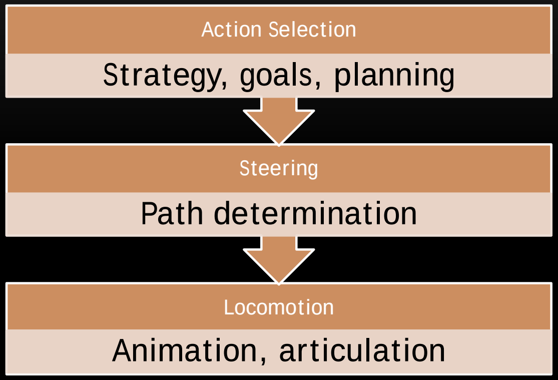
- Leader Following
- Walk Along [10]
  - Two friends go together to a certain place



- Other…?

WALK ALONG - DEMONSTRATION

ACTION SELECTION LAYER



- Which steering behavior should be active?
- Parameters?
- Should be controlled by action selection layer
  - Autonomously vs. Centrally
- Some problems could be solved on the action selection layer
  - Path Following vs. Others
    - Commander and his regiment
    - Detection of being stuck, etc.
  - Setting parameters according to mood, emotions etc.

STEERING BEHAVIORS CONCLUSION

- Advantages
  - Simplicity ➔ predictability (good for debugging)
  - Reactive behavior ➔ efficiency (time, memory)
  - Forces ➔ smoothness, combinability
- Disadvantages
  - Simplicity & Local Traps ➔ low believability ➔ sometimes we need higher-level prediction and planning
  - Scalability (modifying the behavior by hacking extra lines into code)
- Use
  - Computer games, Films
  - Crowd simulations (evacuations, shopping centers, etc.)

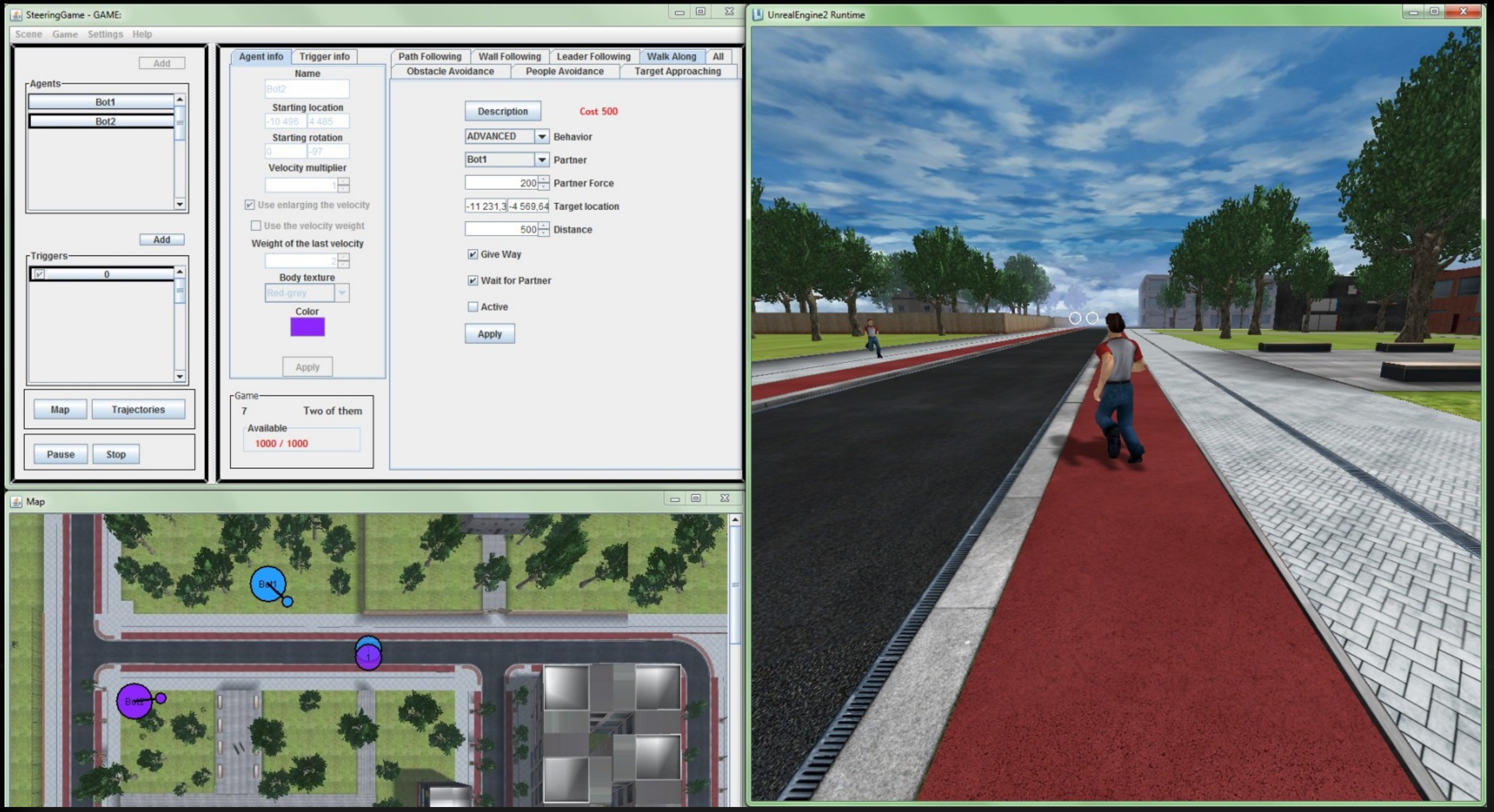
WORKSHOP

- SteeringTool 2.0
- SteeringGame
- UT2004SteeringLibrary

Web & Instalator:

http://diana.ms.mff.cuni.cz/pogamut-games

Project SteeringGame



LITERATURE I.

1. REYNOLDS, Craig W. Flocks, Herds, and Schools: A Distributed Behavioral Model. In Proceedings of Computer Graphics. Anaheim, California : ACM SIGGRAPH, 1987. Pages 25-34. WWW: <http://www.red3d.com/cwr/papers/1987/SIGGRAPH87.pdf>.
2. REYNOLDS, Craig W. Steering Behaviors For Autonomous Characters. In Proceedings of Game Developers Conference. San Francisco, California : Miller Freeman Game Group, 1999. Pages 763-782. WWW: <http://www.red3d.com/cwr/papers/1999/gdc99steer.pdf>.
3. REYNOLDS, Craig W. Steering Behaviors For Autonomous Characters [online]. September 5, 1997 , June 6, 2004 [cit. 2011-05-19]. Steering Behaviors For Autonomous Characters. WWW: <http://www.red3d.com/cwr/steer>.

Related works, Benchmark for Steering Behaviors, Collision Avoidance Model

LITERATURE II.

4. CHAMPANDARD, Alex J. AI Game Development: Synthetic Creatures with Learning and Reactive Behaviors. First printing. United States of America : New Riders Publishing, 2003. ISBN 1-5927-3004-3.
5. CHAMPANDARD, Alex J. AI Game Programming Wisdom 2. First Edition. United States of America : Charles River Media, 2004. An Overview of Navigation System, Pages 131-139. ISBN 1-58450-289-4.
6. SINGH, Shawn, et al. Watch Out! A Framework for Evaluating Steering Behaviors. In Motion in Games : First International Workshop, MIG 2008 Utrecht, The Netherlands, 2008 Revised Papers. Germany : Springer-Verlag, 2008. Pages 200-209. ISSN 0302-9743.
7. KARAMOUZAS, Ioannis, et al. A Predictive Collision Avoidance Model for Pedestrian Simulation. In Motion in Games : Second International Workshop, MIG 2009 Zeist, The Netherlands, 2009 Proceedings. Germany : Springer-Verlag, 2009. Pages 41-52. ISSN 1867-8211.

LITERATURE III.

8. POPELOVÁ, Markéta; BÍDA, Michal. Steering techinky pro virtuální agenty. In KELEMEN, Jozef; KVASNIČKA, Vladimír; POSPÍCHAL, Jiří. Kognice a umělý život XI. Opava : Slezská univerzita v Opavy, 2011. Pages 207-212. ISBN 978-80-7248-644-1.
9. POPELOVÁ, Markéta. Knihovna steering technik pro virtuální agenty. Bachelor thesis. Charles University in Prague, 2011. WWW: <http://amis.mff.cuni.cz/emohawk/> (8.1.2012).
10. POPELOVÁ, Markéta, et al. When a Couple Goes Together: Walk Along Steering. In Proceedings of Motion in Games, Lecture Notes in Computer Science. Volume: 7060, Springer, Heidelberg, Pages 278-289, ISBN 978-3-642-25089-7, 2011.