群组动画
Crowd and Group Animation

金小刚

Email: jin@cad.zju.edu.cn

浙江大学CAD&CG国家重点实验室

紫金港校区蒙民伟楼512


alamy stock photo
HE007C
www.alamy.com


© Francis Pérez

演示——罗马广场的Mumuration现象

椋(liáng)鸟

演示——狮子王

群组的定义

- 在同一物理环境下拥有相同目的的一群个体，他们的行为有别于作为单独个体时的行为。(A large group of individuals in the same physical environment sharing a common goal and may act in a different way than when they are alone.) (Benesh, 86 and Roloff, 81)

- **群组层次**
  - 群 —— Crowd behaviors
  - 组 —— Group behaviors
  - 个体 —— Individual behavior
- **注意一些名词的差异**
  - **Crowd:** 一般指人群；
  - **Flock:** (羊或鸟)群;(尤指同类人的)一大群;
- **粒子系统中的**个体没有智能。而群组中个体是智能体(Agent)。

为什么需要群组动画?

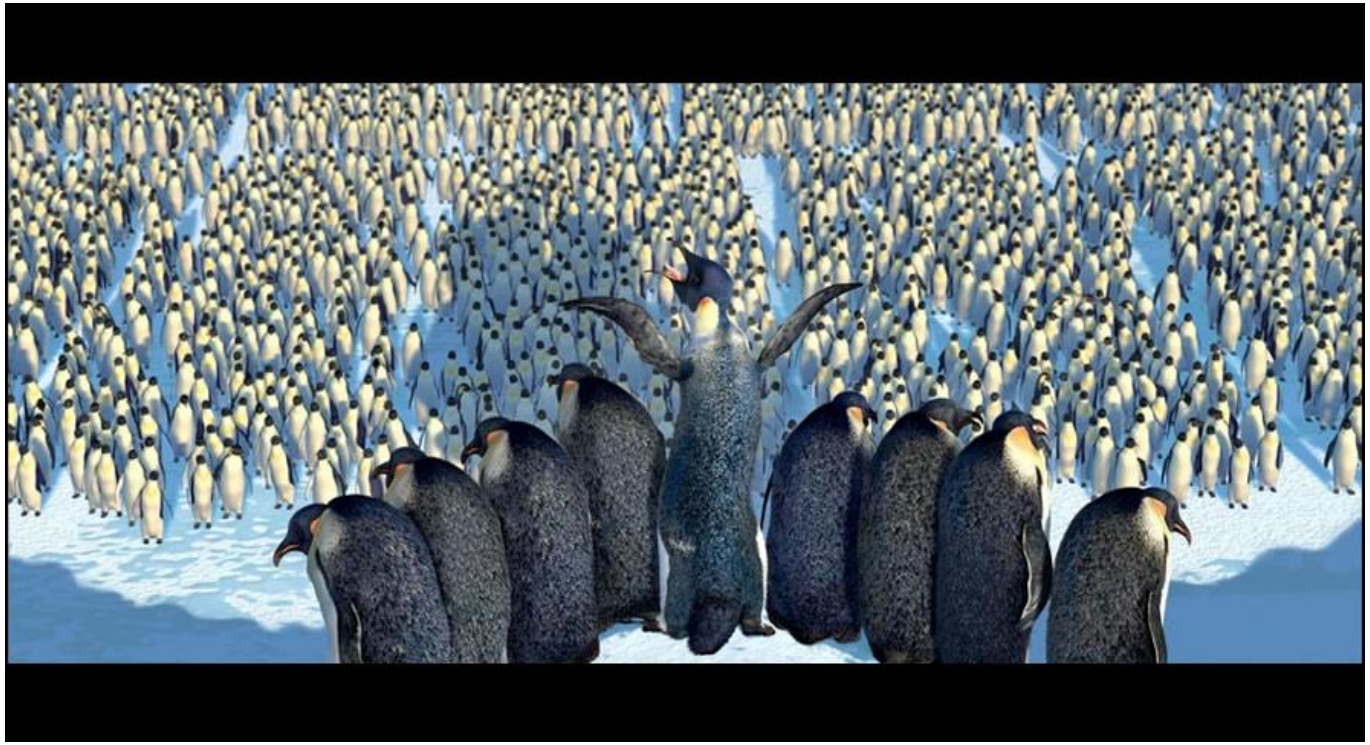
- 得到大场面的震撼效果
  - 动画设计和影视特技中，不可避免地会遇到各种大规模群体动作场面的制作。例如，两军对垒中的数十万大军冲锋的效果、兽群、鸟群等。
- 减少动画师的工作量
- 节约成本
  - 请群众演员成本高、指挥调度难、拍摄困难。

- 成功例子：《指环王》系列、《2012》、《阿凡达》、《长城》、 《僵尸世界大战》(World War Z)、《猩球崛起3:终极之战》 等

群组的特点

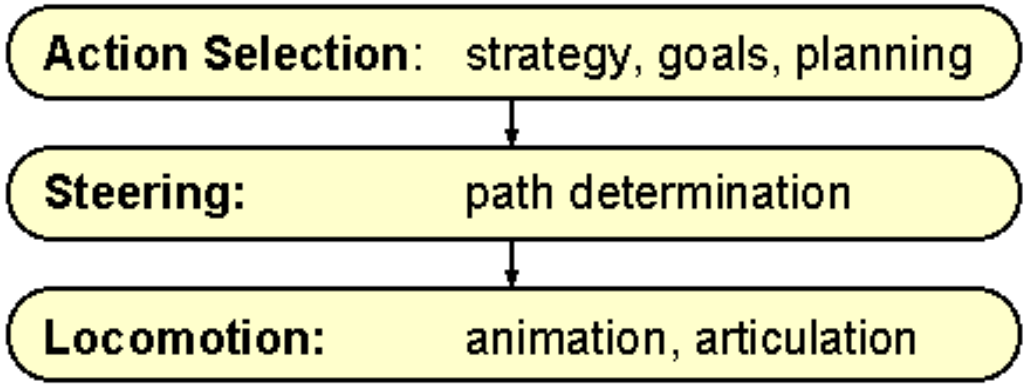
- 数据量大，计算量大，随机性和规律性兼备

- 提出了相应的问题
  - **运动控制**
  - 绘制(Rendering)
  - 创作(Authoring)
  - 建模、LOD等
  - 碰撞检测和处理
  - …



Flocking(群体) System

- 三种层次的行为



将各种导航行为抽象出来，并加以全面地研究和实现。

- **Action Selection:** 基于人工智能的行为选择
- **Steering:** 抽象化的导航行为
- **Locomotion:** 实体化的个体行为
  - 有腿的运动个体：
    - 平衡性，骨骼，肌肉
  - 有轮的运动个体：
    - 发动机，方向盘，刹车

群体模拟软件

- **Atoms Crowd**
  - ToolChefs公司，完全独立的工具，可以容易地集成到任何3D应用程序，并支持集成Maya和Houdini。可以快速生成集群动画，允许在两个3D程序之间轻松改变数据。

- **AXYZ Design Anima**

anima®
The Crowd Simulation Software

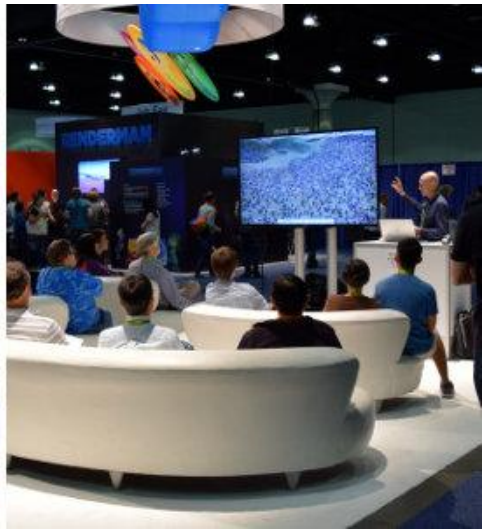
- **GOLAEM**
  - **Population Tools For Maya**

群体模拟软件

- Massive
  - 功能最强大

群组模拟软件Massive

- Weta Digital公司网站http://www.massivesoftware.com/
- 看过“the Lord of the Rings”指环王三部曲的观众都会被其中千军万马的宏大场面所折服，这要归功于能创造数以万计数字角色的群组模拟软件 Massive。



Massive at SIGGRAPH 2015

14 August 2015

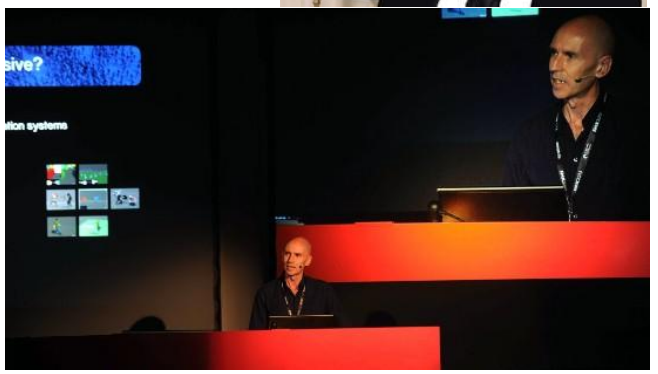
Massive Prime

Massive wins another Emmy(艾美奖，2021)

艾美奖是美国电视界的最高奖项，由美国电视艺术与科学学院主办

群组模拟软件 Massive

- Massive是为指环王三部曲的制作而开发的。
- 其创始人为新西兰人Stephen Regelous
- 当Peter Jackson（指环王三部曲的导演）问Regelous能否为指环王三部曲开发一套群组模拟系统的时候，他立即就被这个项目吸引了。
- Regelous担任了指环王三部曲的群组模拟主管，并花了2年的时间编写出了Massive。
- 荣获2003年奥斯卡科学与工程奖



Massive需要解决的问题

- 要创建出一个具备响应速度快、适应能力强的仿真高品质运动混合引擎
- 要为艺术家设计一个用户界面让他们非常方便的创建人工智能动画
- 设计一个效率极高并且非常稳定的物理模拟引擎
- 要为成千上万的代理数字角色定位

Massive为何能处理庞大数据?

- 专门为处理大规模人群而设计；
- 程序进行了精心优化，以得到最佳的性能；
- 采用了很多内存优化的方法；

群组模拟软件Massive

- Massive是一个先进的自动化角色动画工具。它最初是为了电影指环王三部曲而设计的。在Massive中，数字角色(或称为智能体Agent)对它们所“看到” “听到” “触摸到”的事件自动做出反应。
- 为艺术家使用而设计的，所以并不需要编程。
- Massive是一个完整的产品，包括骨架设计、运动混合、刚体动力学、布料模拟、 Agent设置、位置布局、动作设计、皮肤绑定、材质纹理、灯光工具，并通过经过优化的 RenderMan 进行绘制。

制作的电影


LORD OF THE RINGS


WILL SMITH
I,ROBOT


KING KONG


NARNIA


WORLD TRADE CENTER


NIGHT AT THE MUSEUM


PIRATES OF THE CARIBBEAN
AT WORLD'S END


THE REBELLION BEGINS
Harry Potter


THE GOLDEN COMPASS


10,000 BC


NARNIA
PRINCE CASPIAN


BRENDAN FRASER
JET LI
MUMMY


赤壁


AVATAR

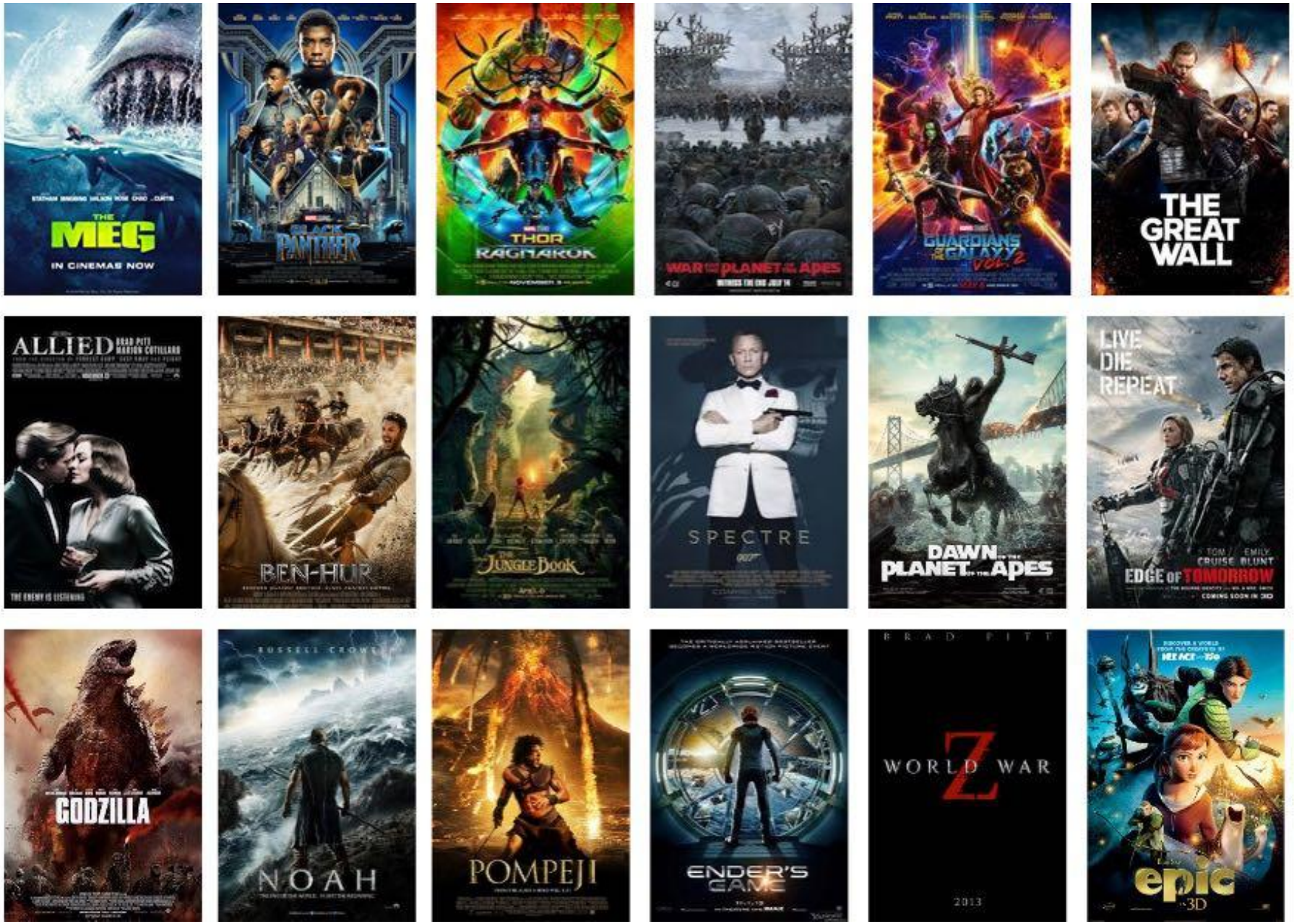

2012


EL SECRETO DE SUS OJOS

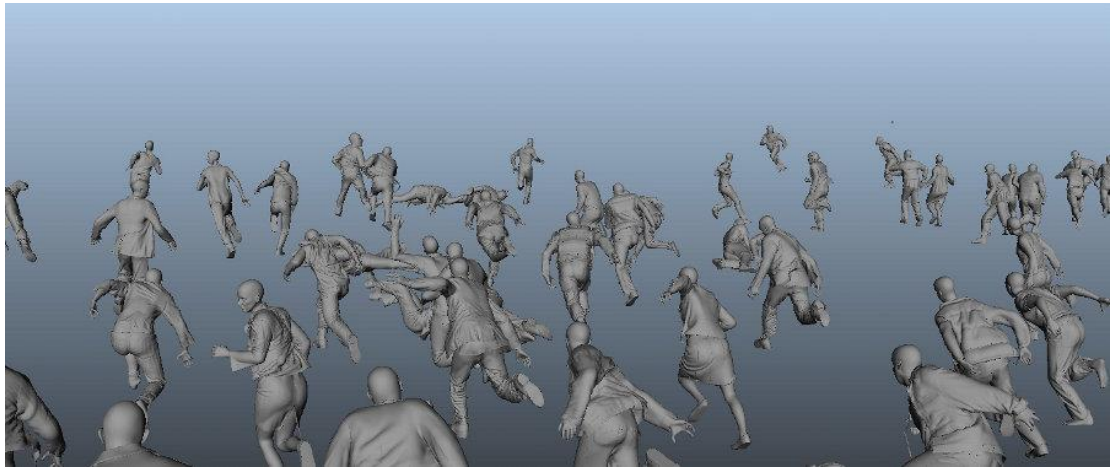

TRON
LEGACY
IN 3D • JANUAR 2011


FINAL DESTINATION

制作的电影


THE MEG
IN CINEMAS NOW
BLACK PANTHER
THOR RAGNAROK
WAR FOR THE PLANET OF THE APES
GUARDIANS OF THE GALAXY VOL. 2
THE GREAT WALL
ALLIED
BEN-HUR
THE JUNGLE BOOK
SPECTRE
DAWN OF THE PLANET OF THE APES
LIVE DIE REPEAT
EDGE OF TOMORROW
GODZILLA
NOAH
POMPEJI
ENDER'S GAME
BRAD PITT
WORLD WAR Z
2013
epic
3D

应用实例——
《World War Z》(僵尸世界大战)》



应用实例——《精忠岳飞》

应用实例——《精忠岳飞》

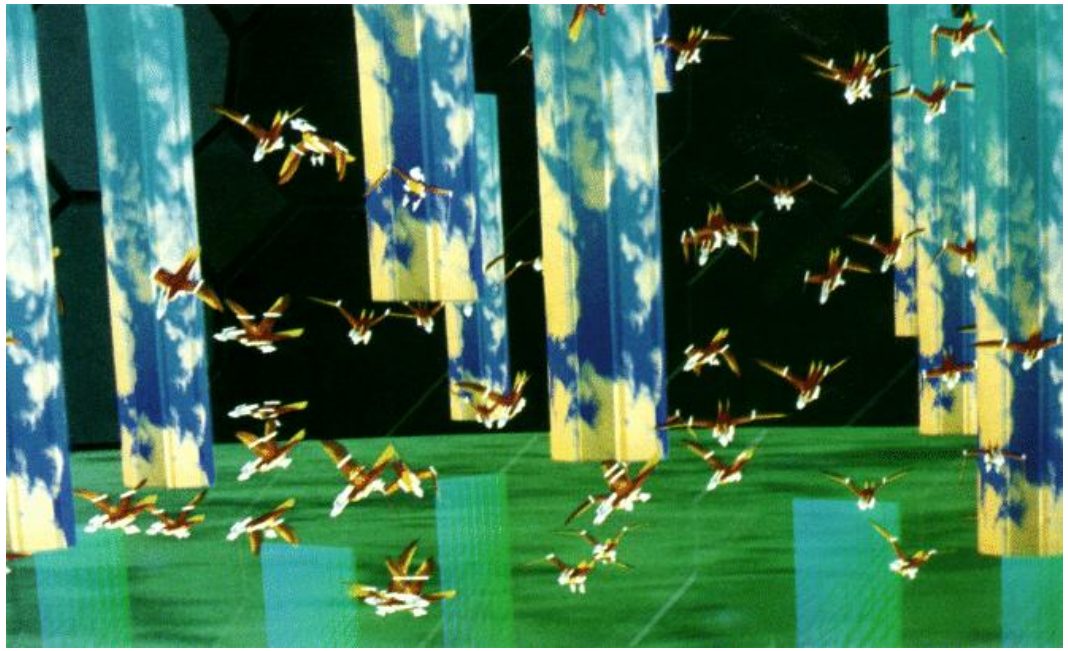
应用实例——《精忠岳飞》

应用实例——《精忠岳飞》

应用实例——《精忠岳飞》

应用实例——《精忠岳飞》

Boids模型



- **参考文献**

  Craig W. Reynolds. 1987. Flocks, herds and schools: A distributed behavioral model. *SIGGRAPH Comput. Graph.* 21, 4 (August 1987), 25-34. (Google citations: 15638次, 2025.10.14)

Flocks and Boids

- Flock 是呈现**对齐**、**非碰撞**、**聚集运动现象**的一组物体 (Flock is a group of objects that exhibit the general class of polarized (aligned), non-colliding, aggregate motion)
- 为了模拟群体现象，我们来模拟一只鸟的行为
  - 感知(Perception)
  - 飞行动力学(Flight dynamics)
- Boid是一个模拟的类似于鸟一样的物体

如何模拟群体行为?

- 用剧本语言：难以描述!
- 粒子系统：太简单。
- 把粒子系统变得更加复杂
  - 引入局部感知(Local perception)
  - 引入物理仿真
  - 呈现群体行为
- 首先模拟一只鸟相对简单的行为，然后通过鸟个体之间大量的互动来呈现群体行为

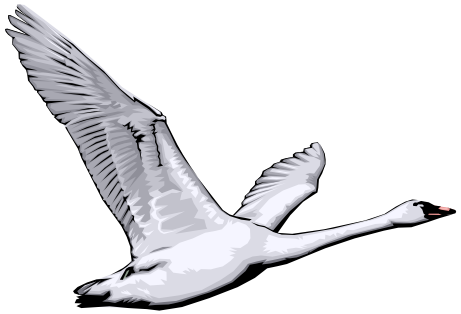
- 动画师作为戏剧导演
- 导演给演员以指示，通过演员的表演来表现演出效果
- 根据指定的行为和初始条件，动画师不知道后续的仿真会如何进行——会出现许多意想不到的惊喜。
- **这些boids似乎有自己想法!**

几何飞行

- 飞行——沿着一条路径进行动态、增量和刚性运动

- 物体沿着一条三维曲线运动，并同时进行几何变换
- 物体是刚性运动，但是底层的几何模型在飞行坐标系中可随意改变形状（如鸟拍翅膀）

- 动量守恒
- 不超过最大速度
- 可指定最小速度



- 最大加速度用来提供平稳的速度变化和航向

几何飞行——倾斜(Banking)

- 考虑重力，但只用力定义**倾斜行为**(banking behavior)
- 很多物理力并不支持

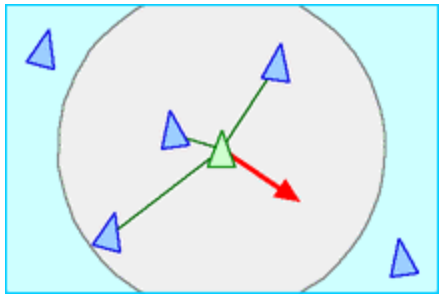
- 转向加速度的大小直接随着运动对象的速度和其路径的曲率而变化
- 正确的倾斜——物体的局部空间与感知坐标系保持对齐
- 使得boid的运动和定方位符合观众的期望

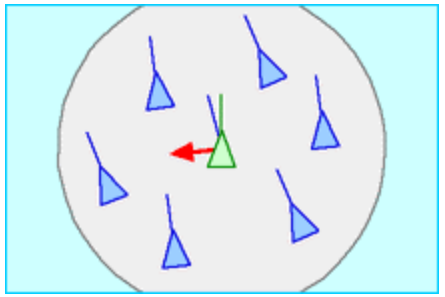
自然的羊群、牛群、鱼群和鸟群

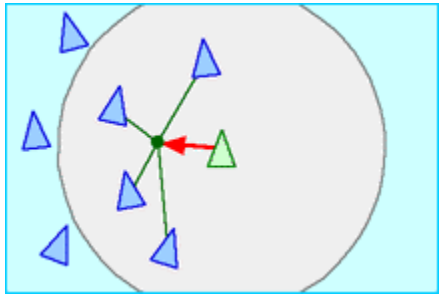
- 鸟群中的鸟儿必须允许它与同伴进行运动协调
- 两个平衡，但相反的行为
  - 与鸟群靠近的欲望
  - 与其它鸟不碰撞
- 一只鸟不会把注意力放在鸟群中的所有鸟上
  - 鸟对鸟群中其它鸟的感知通常是局部和过滤过的
    - 它自己
    - 它周围的邻居

优先级递减的群体模拟三大原则

- **碰撞避免原则**(Collision Avoidance)
  - 避免与相邻的群体成员相碰
- **速度匹配原则**(Velocity Matching)
  - 尽量保持与周围邻居群体成员的速度匹配
- **群体合群原则**(Flock Centering)
  - 群体成员尽量靠近

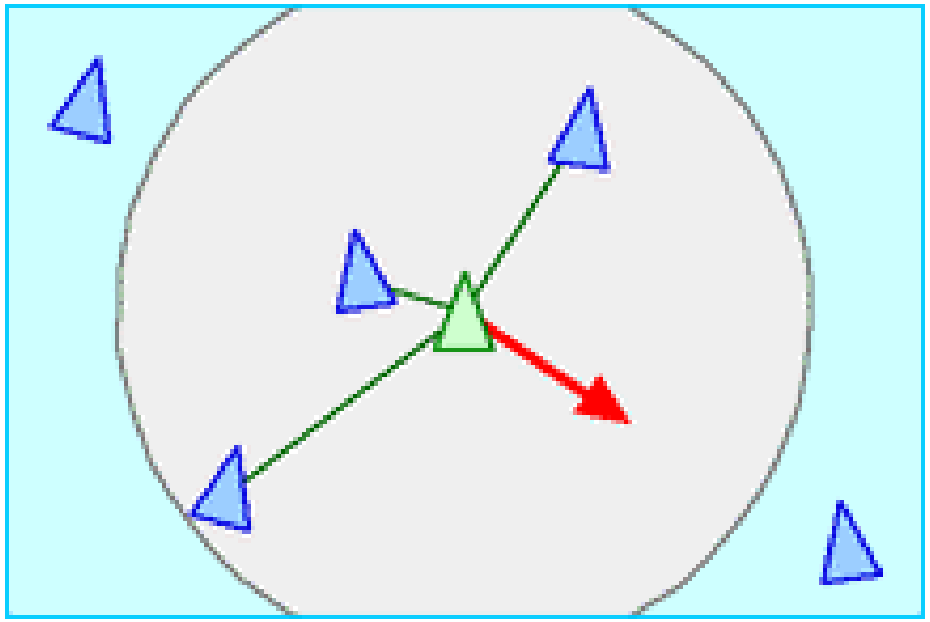


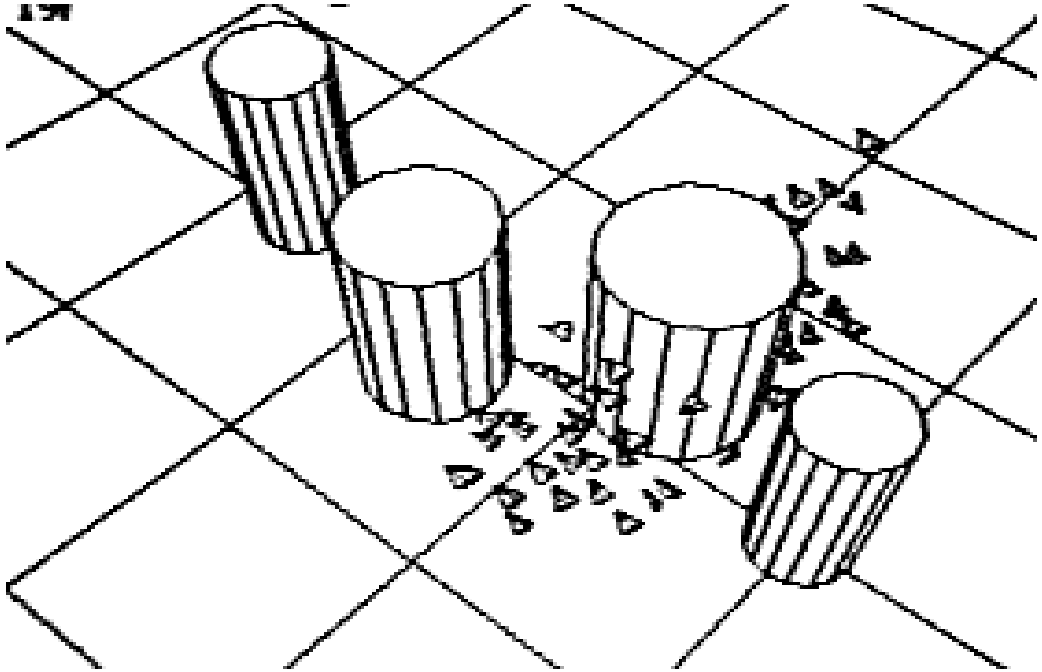




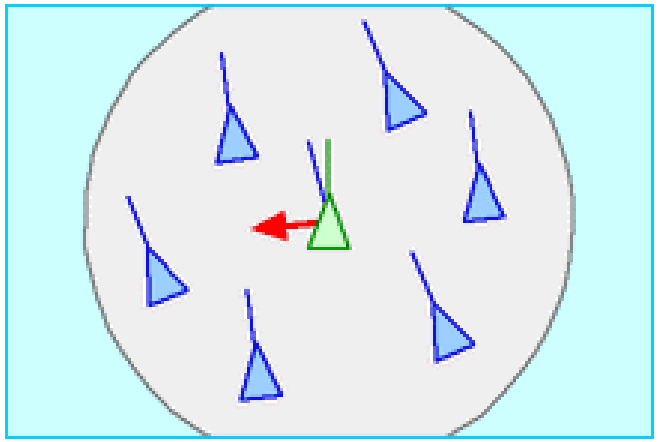
碰撞避免

- 努力摆脱迫在眉睫的影响
- 基于群体成员相对位置的静态碰撞避免
- 忽略速度



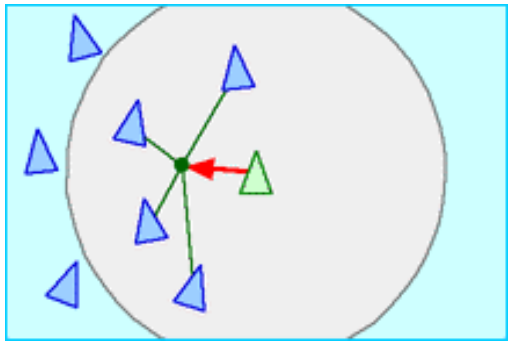


速度匹配原则(Velocity Matching)



- 只考虑速度
  - 忽视位置
- 碰撞避免预测版
  - 如果boid与相邻群体成员的速度匹配得很好，则将来不大可能发生碰撞
- 静态碰撞避免和动态速度匹配是相辅相成的
- 静态碰撞避免建立所需的最小分离距离，而速度匹配则尽量保持这个距离

群体合群原则(Flock Centering)

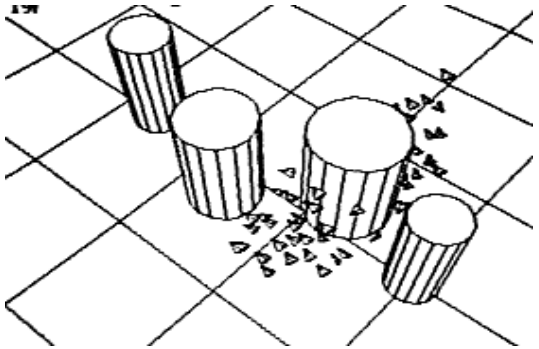


- Boid局部感知的群体中心，实际上是相邻群体成员的中心；
- 使得boid飞向与相邻boids的中心；
- 如果一个boid接近群体的中心，该原则对它的影响不大（因为boid的密度将变均匀）；但如果它处于群体的边缘，这将对它有很大的影响；
- 允许模拟群体分裂和合并

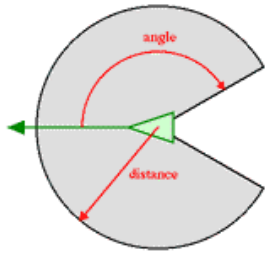
仲裁独立的行为

- 这三个行为表现为广义的加速请求
- 行为有多个参数，包括强度，进一步对加速请求进行调配
- boid大脑的导航模块收集相关的加速请求，然后确定单一行为所需的加速
- 根据优先级进行加权平均
- 因为加权平均会取消相反的方向，在处理与障碍物的碰撞时，效果不好

- 根据优先级次序，把加速请求加入到累加器
- 强度添加到另一个累加器
- 继续执行，直到累加的强度的和大于最大加速度值
- 紧急加速先进行分配，以满足迫切需要
  - 例如: 忽略合群原则以回旋绕过障碍物



感知模拟



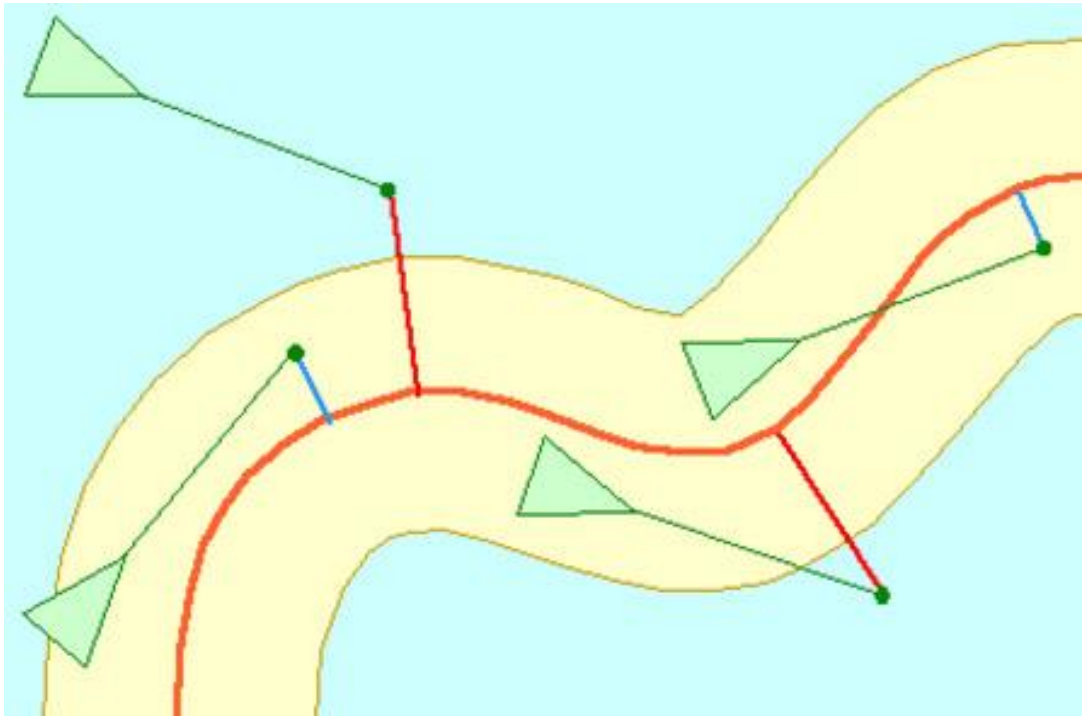
- 感知模型试图模拟真实的鸟
- 过滤掉过剩的信息，实现boid的行为
- 对于一只真实的鸟，其感知是不完美的。邻近的群组成员将遮挡远处的群组成员
  - 只知道局部的信息
- 聚集的群体运动依赖于有限的、局部的视野
- 每个boid可以获得整个场景的几何描述，但是它只对周围的小部分邻居群组成员进行响应；
- 邻居群组成员由一个距离 (从boid的中心开始度量)和一角度(从boid的飞行方向开始度量)来刻划；
- 不在该范围的群体成员将被忽略；
- 在该区域内的群体成员将影响boid的导航；

Flocking的整体控制

- 整体目标，全局方向
- 整体位置——鸟朝着这一个目标点飞行
- 有界加速，使得boid逐步飞向迁徙目标
- 对整体位置矢量、全局方向矢量设置动画
- 使得目标点沿着给定的路径变化

Steering Behaviors for Autonomous Characters

总结了自主角色的大部分导航行为



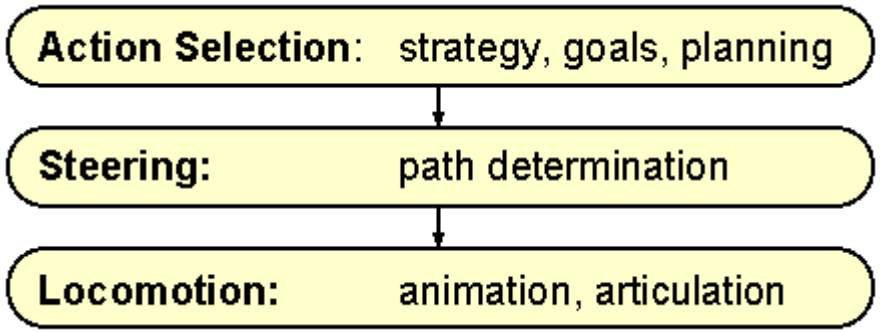
- 很多游戏引擎采用了这些技术：
  如UNITY 3D，Cocos2d-x等。

- **参考文献**

Craig W. Reynolds, Steering Behaviors For Autonomous Characters, *Game Developers Conference 1999*. (Google citations: 2264, 2025.10.14)

背景和思路

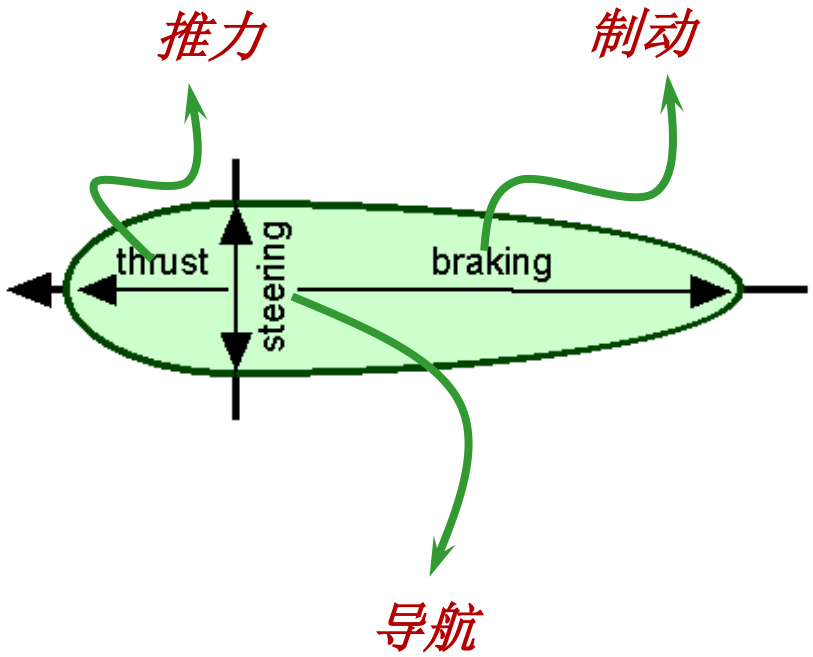
- 在游戏和动画中，如何对**自主角色**进行栩栩如生和即兴的方式进行导航?
- 运动行为的层次结构



- 简单的车辆模型(Simple Vehicle Model)
- 模型的物理原理
- **导航行为**
  - **一个或两个角色的行为**
  - **群组行为**
- **行为的结合**

简单的车辆模型(Simple Vehicle Model)

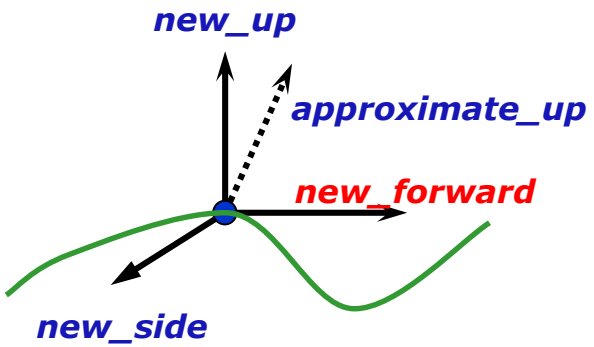
- Simple Vehicle Model
  - mass scalar
  - position vector
  - velocity vector
  - max_force scalar
  - max_speed scalar
  - Orientation *N* basis vectors



车辆模型的物理原理

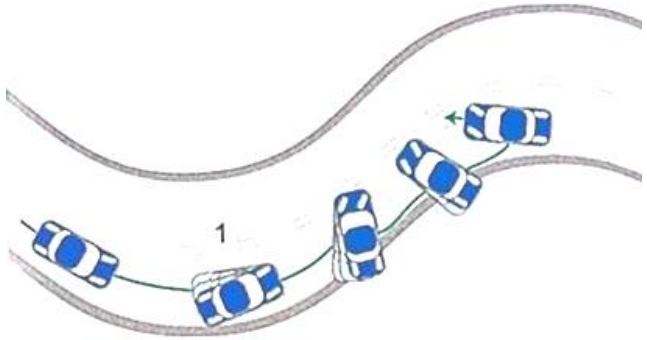
- 简单车辆模型的物理原理为前向欧拉积分法：
  - **steering_force** = truncate (steering_direction, max_force)
    acceleration = steering_force / mass
    velocity = truncate (velocity + acceleration, max_speed)
    position = position + velocity
- 构造新的基向量（局部坐标系）：
  - **new_forward** = normalize (velocity)
    approximate_up = normalize (approximate_up) // if needed
    new_side = cross (new_forward, approximate_up)

    new_up = cross (new_side, new_forward)

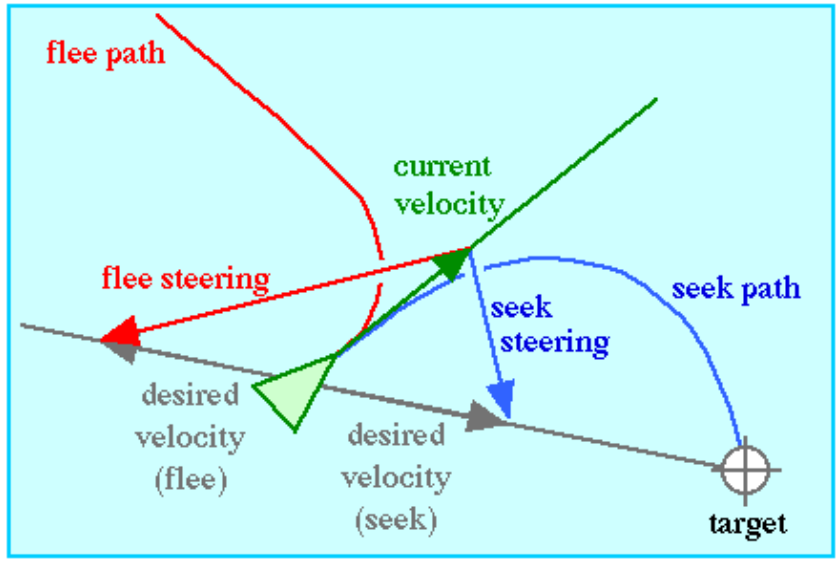


限制(Constraint)

- 这个简单的车辆模型不能模拟如打滑、自旋等效果；
- 而且，该模型允许车辆在速度为零时转弯。

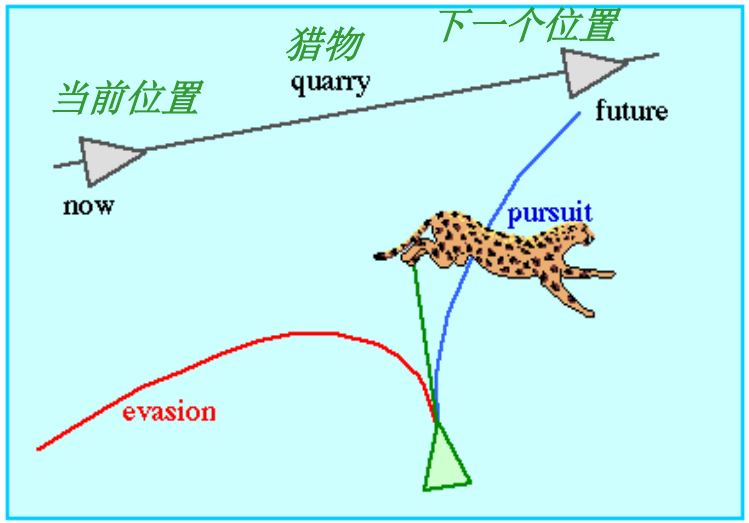


寻找和逃离(Seek and Flee)



- 寻找(或追逐)一个静态的目标
  - **desired_velocity** = normalize (target - position) * max_speed
    steering = desired_velocity – velocity
  - 把追逐反过来，即为逃离
- http://www.red3d.com/cwr/steer/SeekFlee.html

追逐和躲避(Pursuit and Evasion)



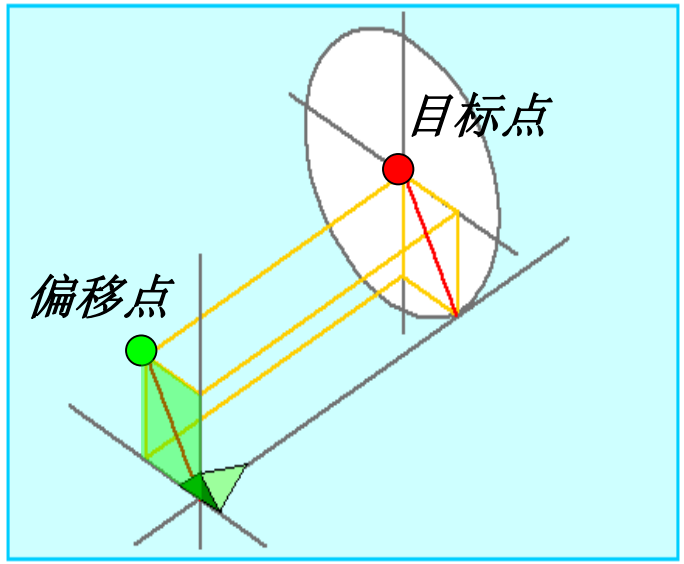
- 追逐(Pursuit)与寻找(Seek)非常类似，其区别在于目标为一移动的角色(猎物)。
  - 假设猎物在预测区间T内不会转向。
  - 猎物在将来的位置可通过把它的当前速度乘以T，并把该偏移量与当前位置相加来得到。
- http://www.red3d.com/cwr/steer/PursueEvade.html

追逐和躲避(Pursuit and Evasion)

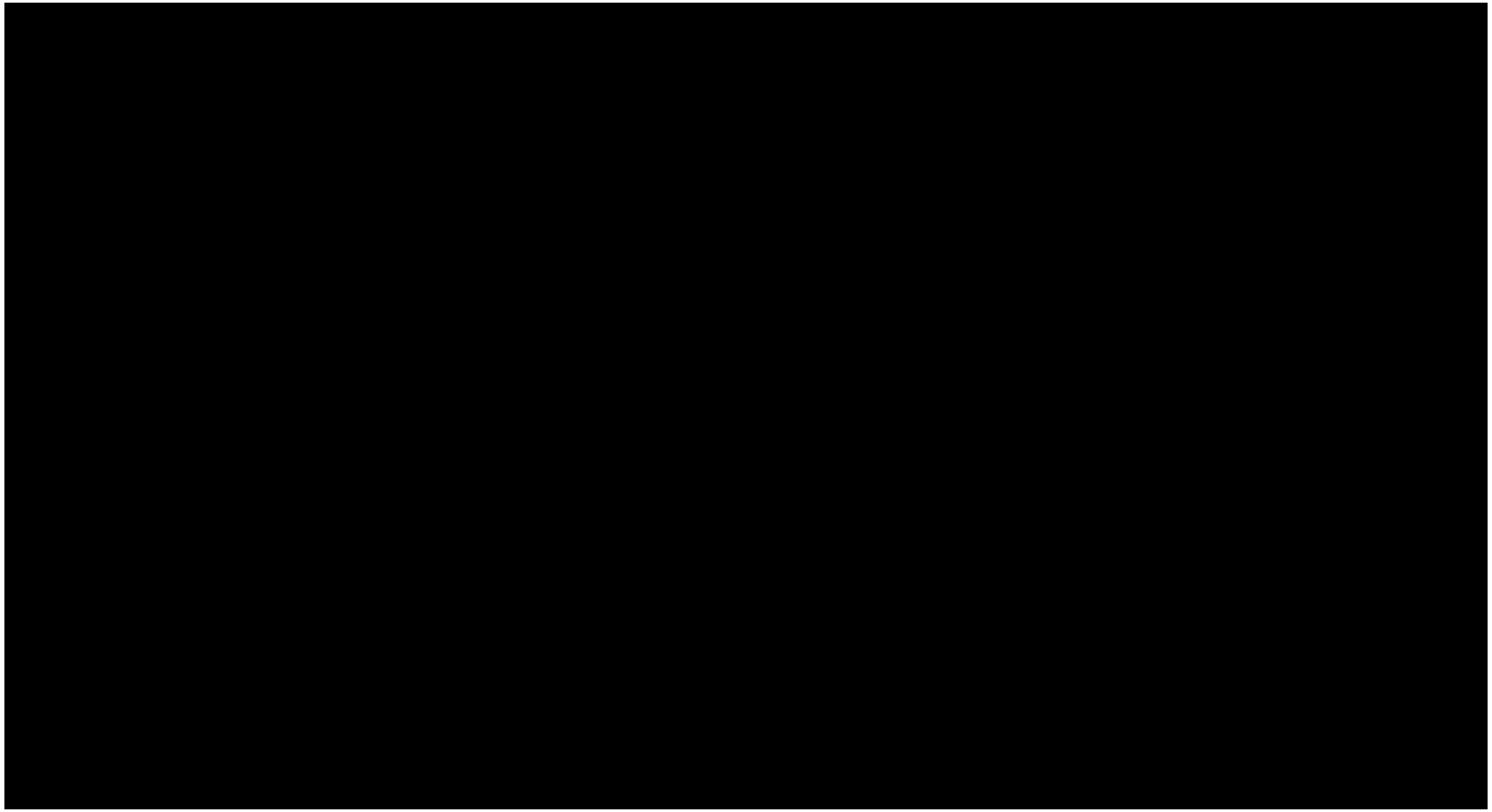
注：也有把Pursuit and Evasion说成Seek and Flee的

偏移的追逐(Offset Pursuit)

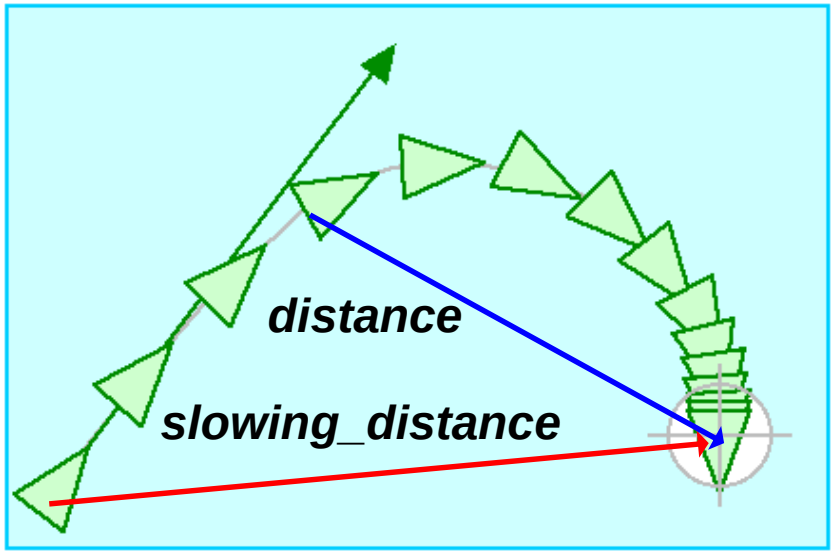
- 沿着一条接近目标的路径追逐，但是不直接进入移动的目标（例子，战斗机进行空中扫射）；
- 对预测的目标点进行局部化(在目标角色的局部坐标系)，把局部目标投影到角色的侧向(Side-Up)平面，对偏移向量进行归一化，乘上比例因子R(偏移值)，并把它加到 局部目标点, 然后把该值进行全局化(变换回世界坐标系)。
- 采用寻找(Seek)行为来接近偏移点。



偏移的追逐(Offset Pursuit)

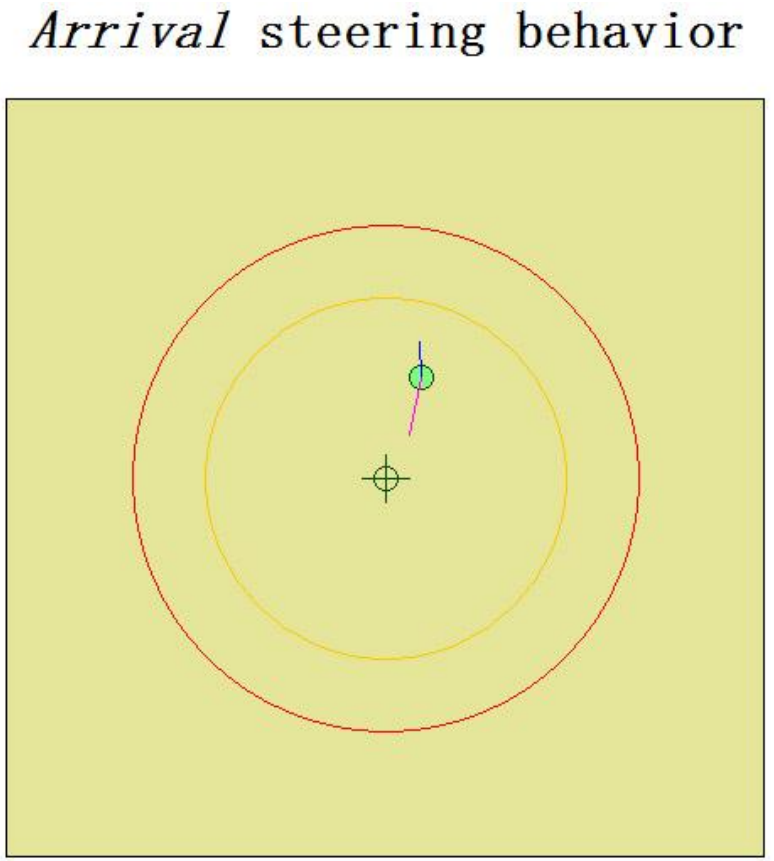


到达(Arrival)



- 当角色远离目标时，到达行为与寻找行为几乎是一样的；
- 但是，寻找行为是以全速穿过目标。而对于到达行为，角色将放慢速度，直到停止，并与目标位置重合。
- 开始减速的距离为到达行为的一个参数；

到达(Arrival)



到达(Arrival)

- http://www.red3d.com/cwr/steer/Arrival.html
- **target_offset** = target – position **(这是一个矢量)**

  distance = length (target_offset) **(这是一个标量)**

  ramped_speed = max_speed * (distance / slowing_distance)

  clipped_speed = minimum (ramped_speed, max_speed)

  desired_velocity = clipped_speed * **target_offset** / distance

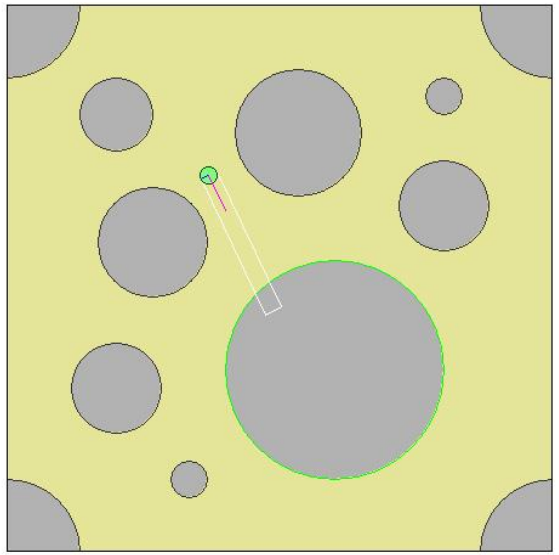
  steering = desired_velocity - velocity

slowing_distance: 表示开始减速的距离

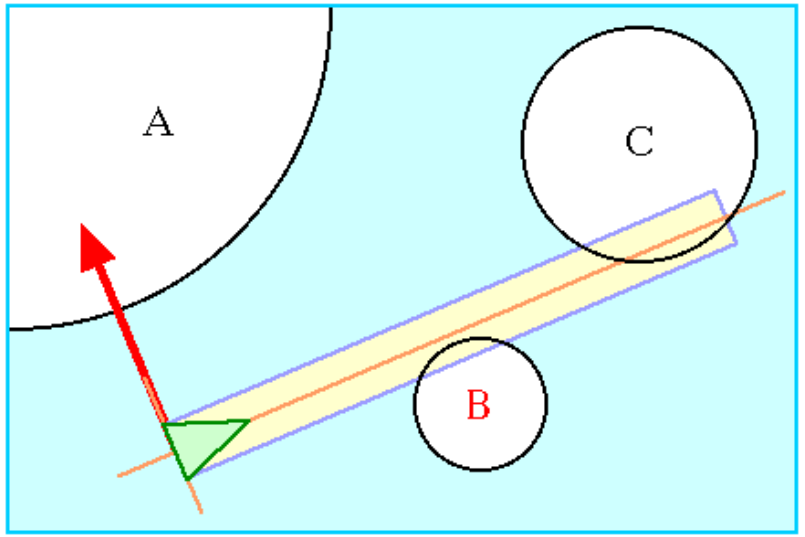
到达(Arrival)

障碍避免(Obstacle Avoidance)

Obstacle Avoidance steering behavior



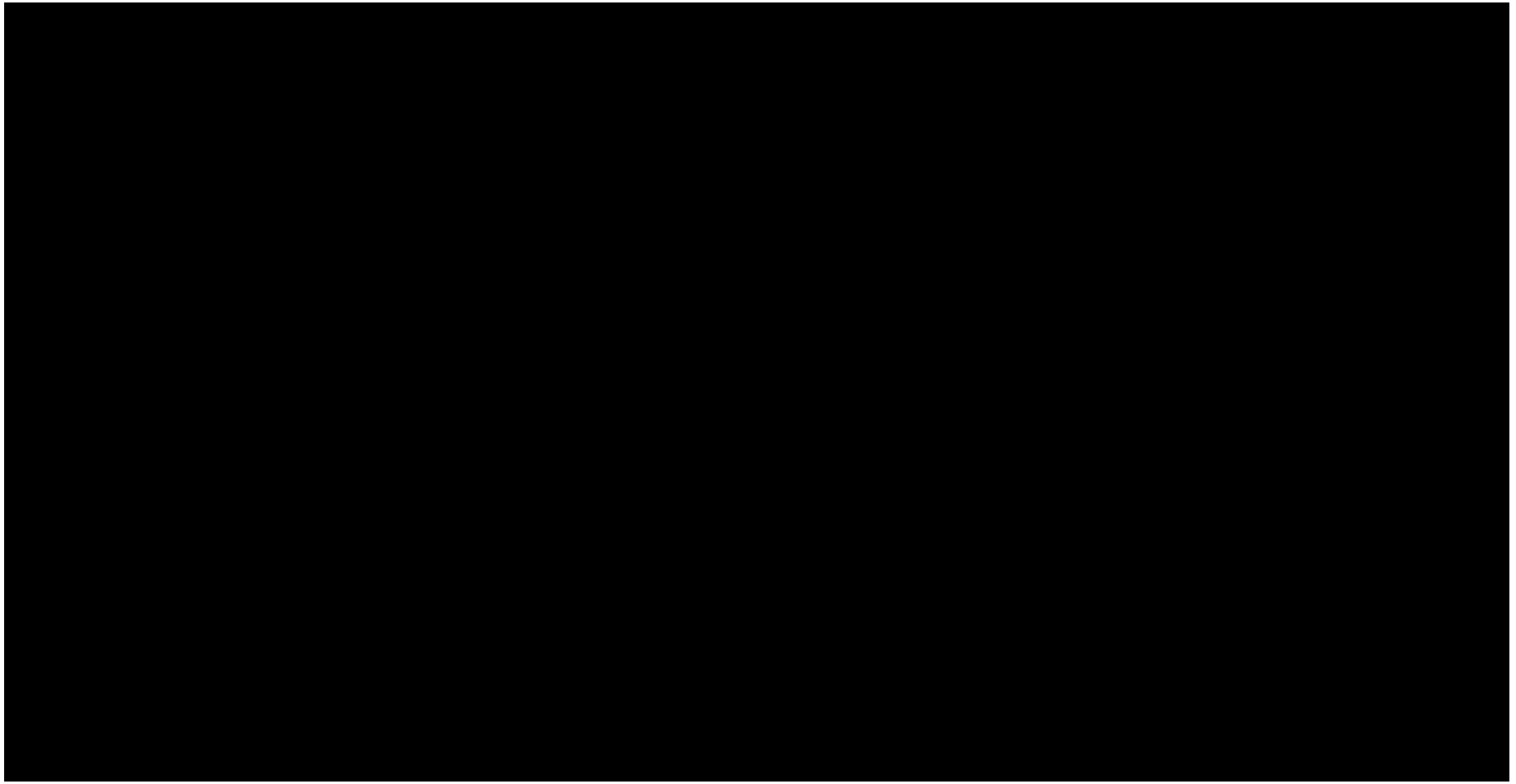
anticipatory collision avoidance



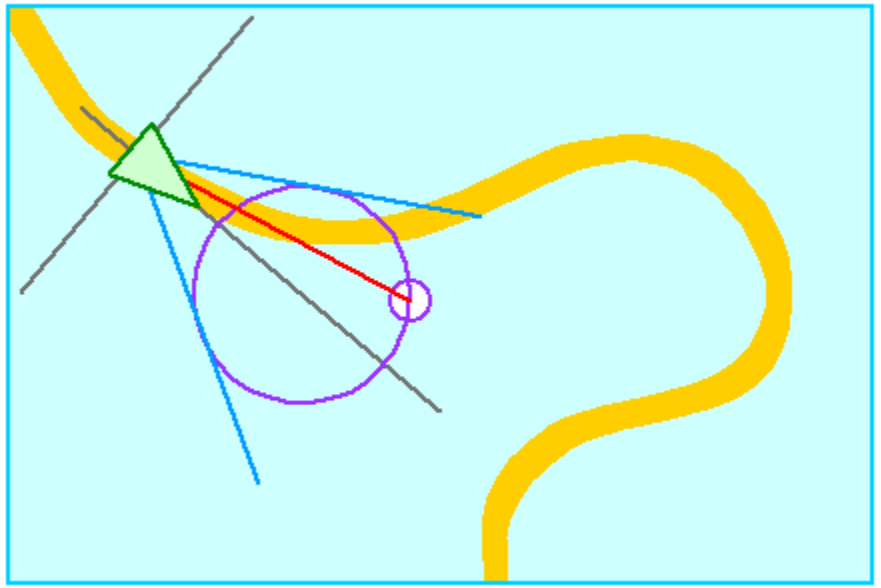
- 该行为的目标是：在该角色前面，保证有一个假想圆柱的自由空间(实际上是预期性碰撞避免)。
- http://www.red3d.com/cwr/steer/Obstacle.html

- 障碍避免返回的值：
  - 返回避免最有威胁障碍物的导航值；
  - 如果没有碰撞是迫在眉睫的，返回一个特殊值（空值，或零向量），表示在这一时刻不需要纠正量。

障碍避免(Obstacle Avoidance)

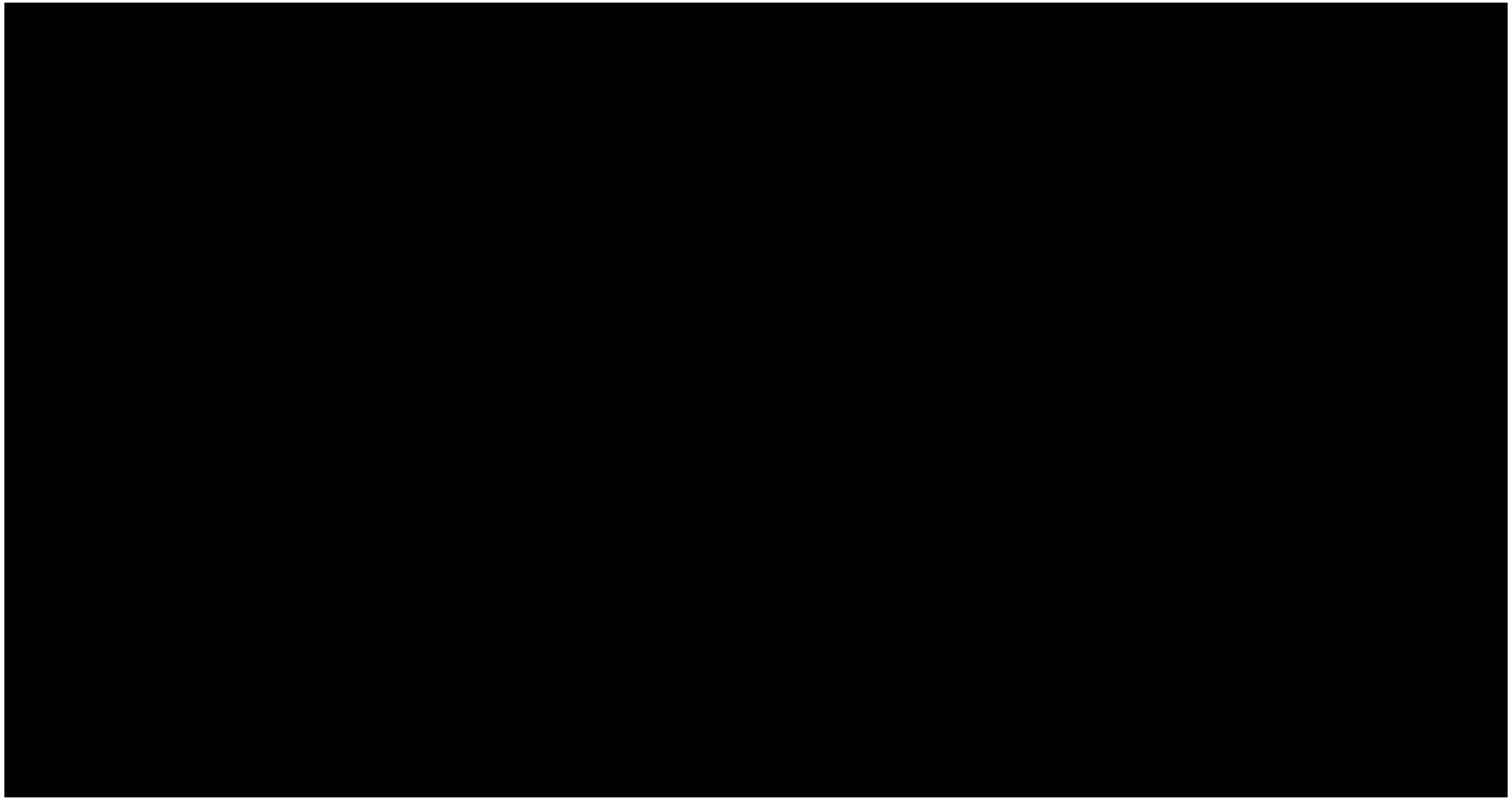


徘徊(Wander)



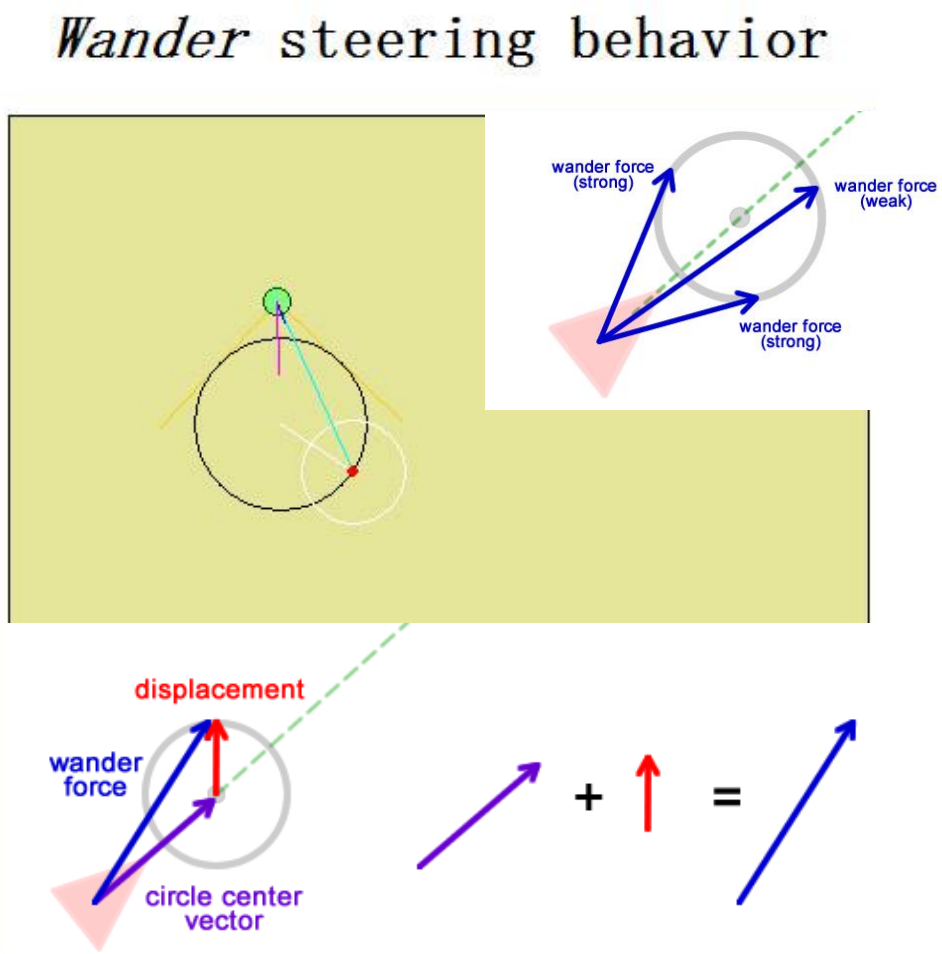
- http://www.red3d.com/cwr/steer/Wander.html
- 导航方向表示为**红色向量**
- 图中的**大圆**对导航进行约束
- **小圆**对导航的随机偏移量进行约束

徘徊(Wander)



徘徊(Wander)

- 一种随机导航，但又有一定的秩序：某一帧的导航方向与下一帧的导航方向有关。这将产生比生成一随机导航方向更有趣的运动(旨在产生逼真的“随意”动作，让玩家认为角色真的还活着并且四处走动）。
- 徘徊行为将保持一个“徘徊方向”这一状态，这里表示为一个红点。调整的徘徊方向约束到一个黑圆上。黑圆的半径为1，其中心处于沿角色向前轴(*forward* axis)sqrt(2)单位长度的位置。
- 徘徊模型的参数:
  - Strength：从0(没有转向) 到 1.
  - Rate：控制 “徘徊方向“变化的程度，从0（没变化）到1 (快速的飘忽不定的变化），这里表示为白圆的半径 (上图显示的为 0.6，表示最大的随机偏移量)
- 导航方向表示为青色的矢量。车辆的速度用品红矢量来表示，导航力用蓝色矢量来表示。



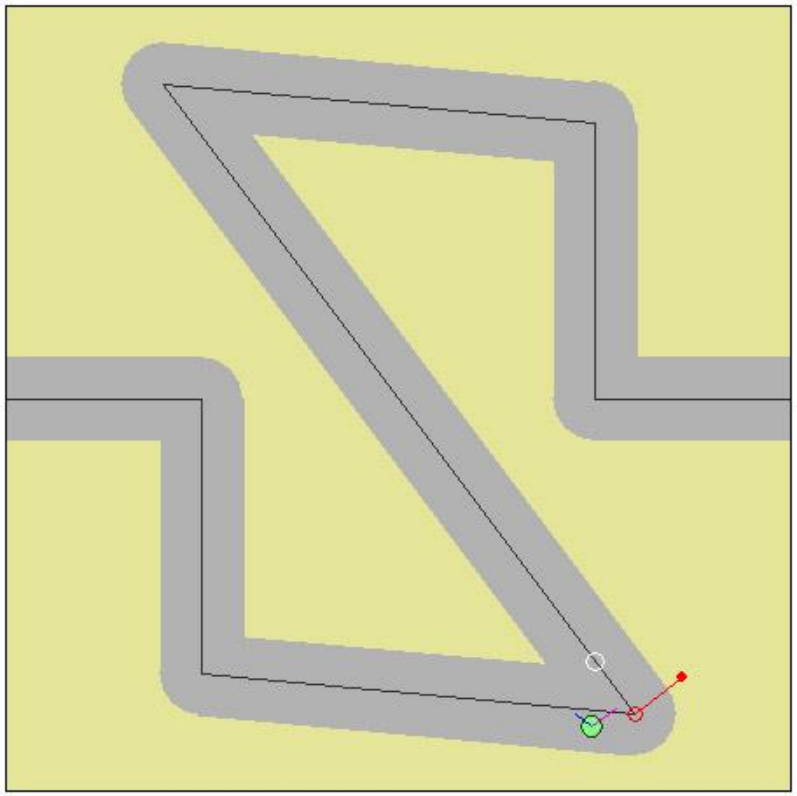
路径跟随(Path following)

- 绿圆显示了路径跟随行为。其任务是在一个给定的方向进行路径遍历（左边进入，右边退出），同时保持其中心位于灰色区域。

- 在该例子中，路径定义为一条折线和一个半径。该行为与抑制(containment)和沿墙运动(wall following) 相关，其区别在于，在路径跟随中，路径意味着行驶方向；

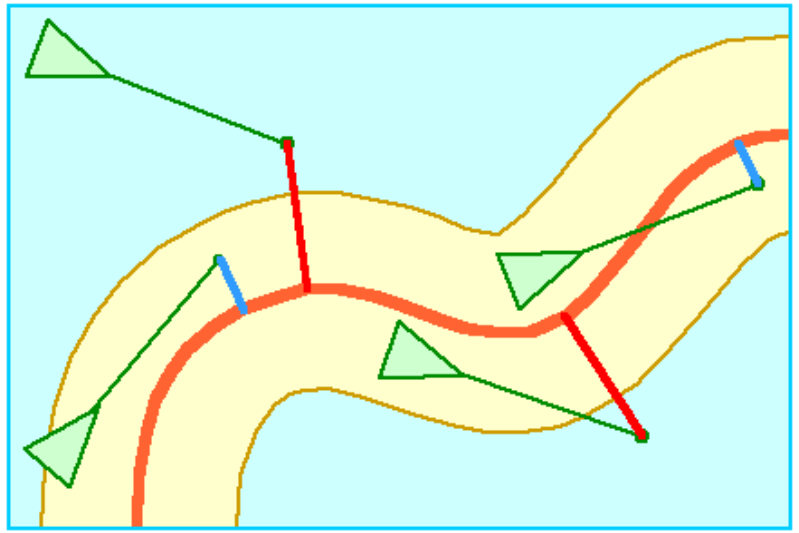
- 在路径跟随行为中，只有当车辆将离开灰色区域时，才执行校正导航。

Path Following steering behavior



路径跟随(Path following)

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



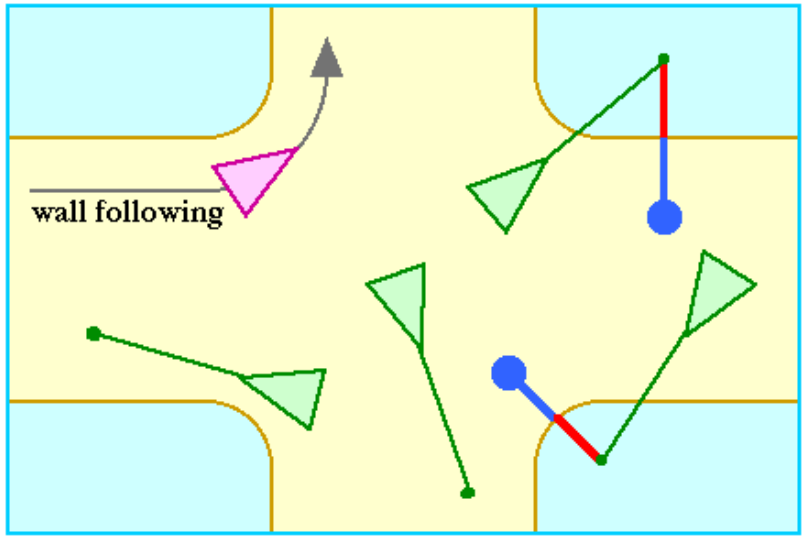
- 沿着一条路径移动角色，并同时保持在脊柱线的指定半径内。
- **投影距离**：小于路径半径时，不需要进行导航校正。
- 否则，把预测的位置投影到路径上，把该点作为目标点，并进行寻找(Seeking)行为。

投影距离(Projection distance)：
从预测位置到最近路径点的距离

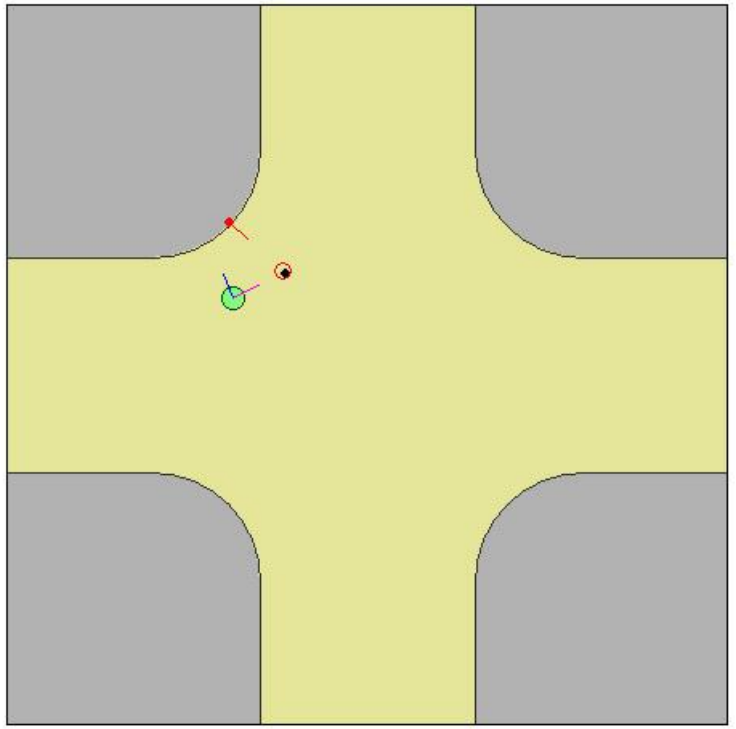
路径跟随(Path following)

沿墙运动(Wall Following)

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



Wall Following steering behavior

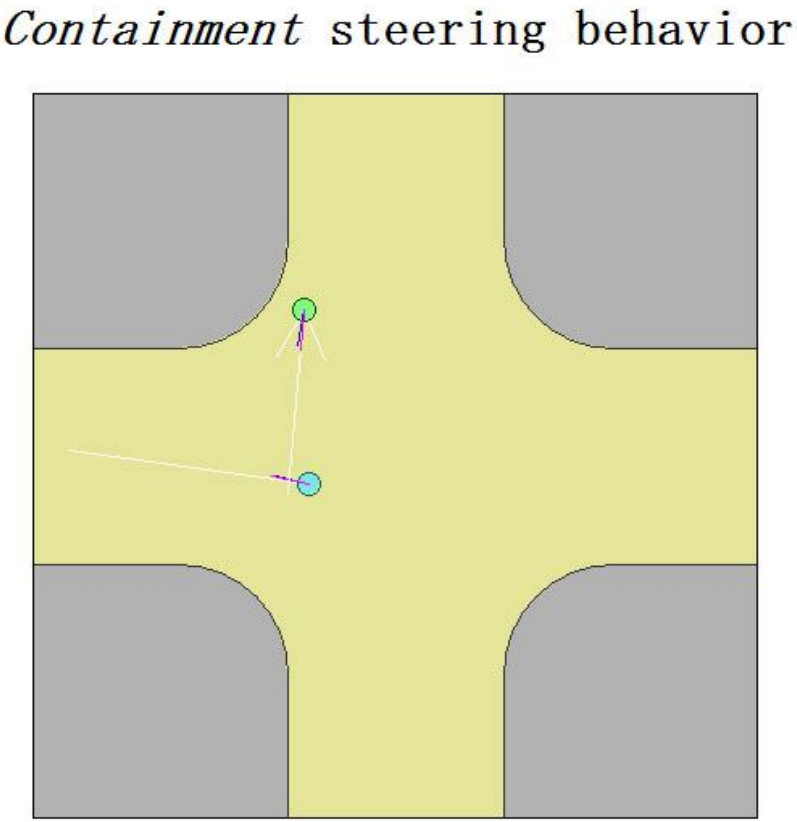


沿墙运动(Wall Following)

- 绿色车辆使用沿墙运动行为以使得它沿标记为灰色的区域平行移动并保持一定的偏移量。汽车在移动时，需要保持与墙给定的距离。车辆的速度由品红色矢量表示，导航力由一蓝色矢量表示。
- 沿墙运动行为的实现：首先根据当前的速度预测一定时间后车辆的位置，把该位置投影到墙上离它最近的点(红点)，把该点沿墙的法向(红线)移动给定的偏移量，从而生成一个目标点(红色的圆)。最后采用寻找(Seek)行为来接近目标点。
- 车辆与墙之间的通信通过一个通用的表面协议(generic surface protocol)来实现：车辆经过墙表面时，经过询问可得到墙上的最近点和该点的法向。因此，导航行为并不需要知道墙壁表面的形状信息。

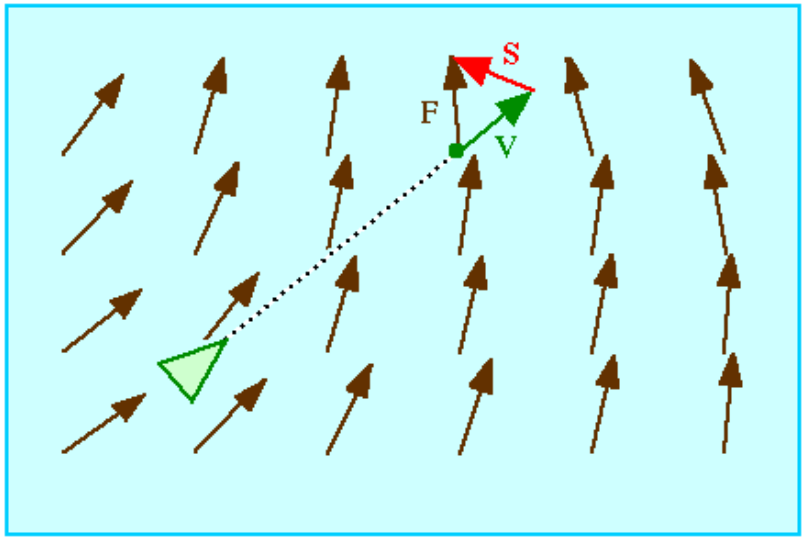
抑制 (Containment)

- http://www.red3d.com/cwr/steer/Containment.html
- 车辆使用抑制(或广义的障碍避免)行为使得它远离标记为灰色的区域。这种行为与前述的障碍避免密切相关（障碍物通过包围球或圆柱表示）。然而，这种行为可以处理任意形状，并允许车辆导航接近障碍物的表面。
- 车辆通过探测点测试前面的空间(用白线表示), 当没有接触障碍物时，车辆将采用徘徊行为，以避免使得它的运动与“通道”对齐。当探测点接触到障碍物时，把该点投影到障碍物的表面，并计算这点的法向，沿法向偏移一定量的点作为Seek目标进行导航。
- 车辆与障碍物之间的通信通过一个通用的表面协议(generic surface protocol)来实现：车辆询问探测点是否位于障碍物内部，如果是的话，得到障碍物上的最近点和该点的法向。因此，导航行为并不需要知道墙壁表面的形状信息。



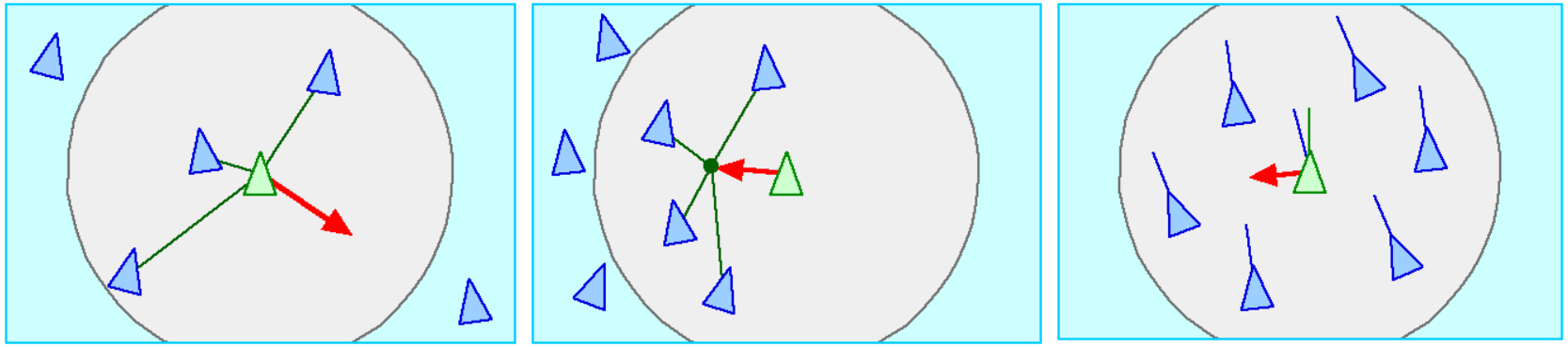
流场跟随行为(Flow Field Following)

- http://www.red3d.com/cwr/steer/FlowFollow.html
- 假设运动区域已经有一个速度流场
- 估算角色将来的位置并计算在这点的流场
- 得到的速度(F)即为我们期望的速度，导航方向(S)为期望速度和当前速度的差



群组的行为(Group Behavior)

- 分离(Separation)、聚集(cohesion)和对齐(alignment)与群体的行为有关；
- 导航行为决定一个角色如何对周围的角色做出反应；

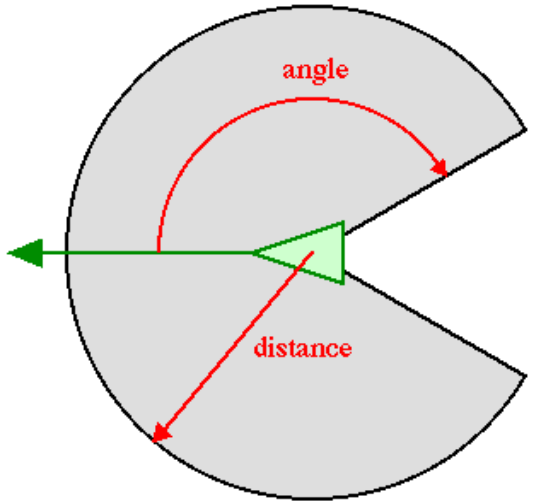


Repulsive force is computed by subtracting the positions of our character and the nearby character

平均速度为我们期望的速度

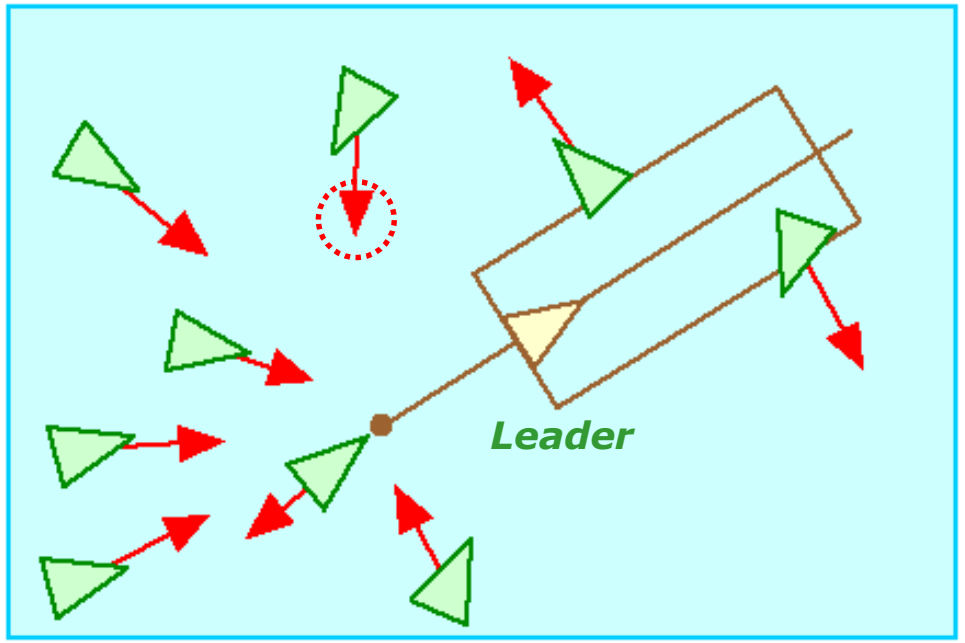
邻居(Neighborhood)

- 角色的局部邻居
  - 邻居群组成员由一个距离 (从角色的中心开始度量)和一角度(从角色的运动方向开始度量)来刻划，该角度定义了角色的感知视野；



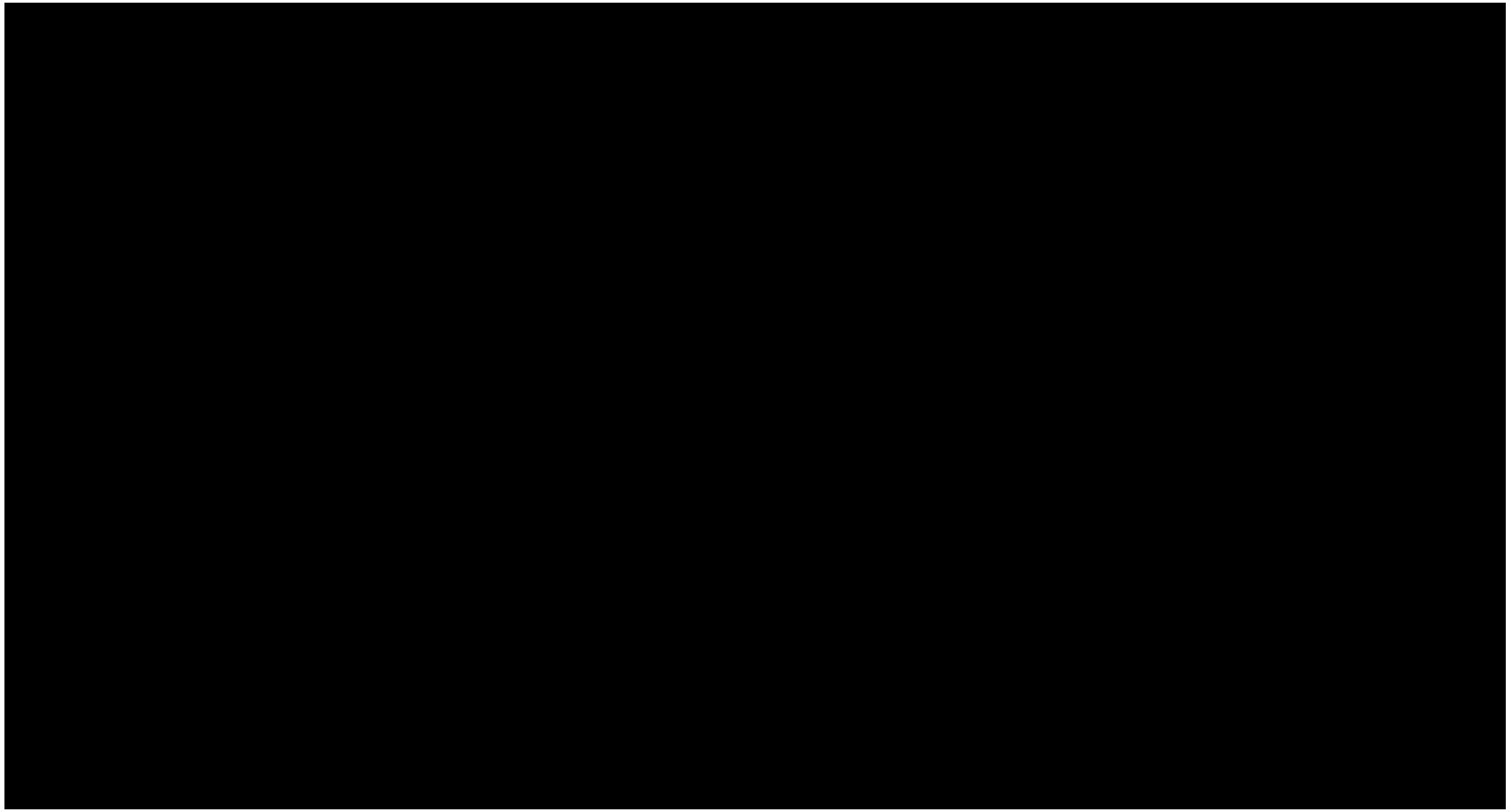
跟随领导(Leader Following)

- 如果一个跟随成员发现自己处于领导前面的一个矩形区域，它会横向远离领导者的路径；
- 否则，到达(arrival)目标为领导后面的一个偏移点；
- 跟随成员采用分离行为来避免相互拥挤；



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

跟随领导(Leader Following)



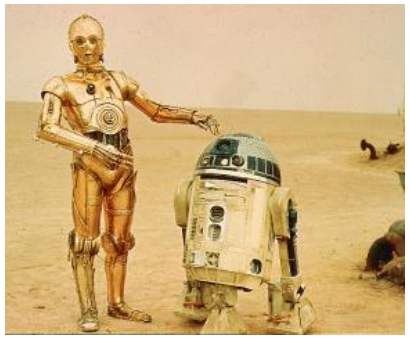
行为的组合(Combining Behaviors)

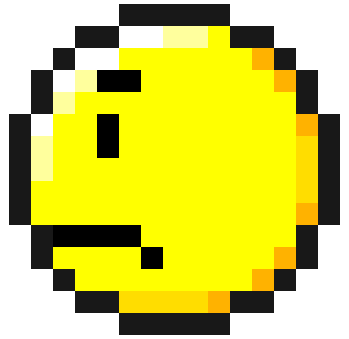
- 组合行为可以采用如下两种方式:
  - 切换(Switch)
  - 混合(Blending)
- 最直接的方法可简单地计算每个组的导航行为并把他们加权求和。

总结

- BOIDS模型：类似鸟集群运动模型（空中）
- Steering Behaviors for Autonomous Characters：地面自主角色模型的导航

- **方法缺陷：**
  - 运动可能会不太自然
    - 运动像一个完美的机器人
  - 碰撞避免后
    - 一遍又一遍沿着障碍物转
  - 碰撞反应
    - 碰撞避免有可能失败





关于Craig W. Reynolds (born March 15, 1953)

- **http://www.red3d.com/cwr/**

- Winner of a Scientific And Engineering Award presented by The Academy of Motion Picture Arts and Sciences for *pioneering contributions to the development of three dimensional computer animation for motion picture production* at the Scientific and Technical Awards of the 70th Academy Awards® held in 1998.

- 1998 to present
  - **Sony Computer Entertainment America**

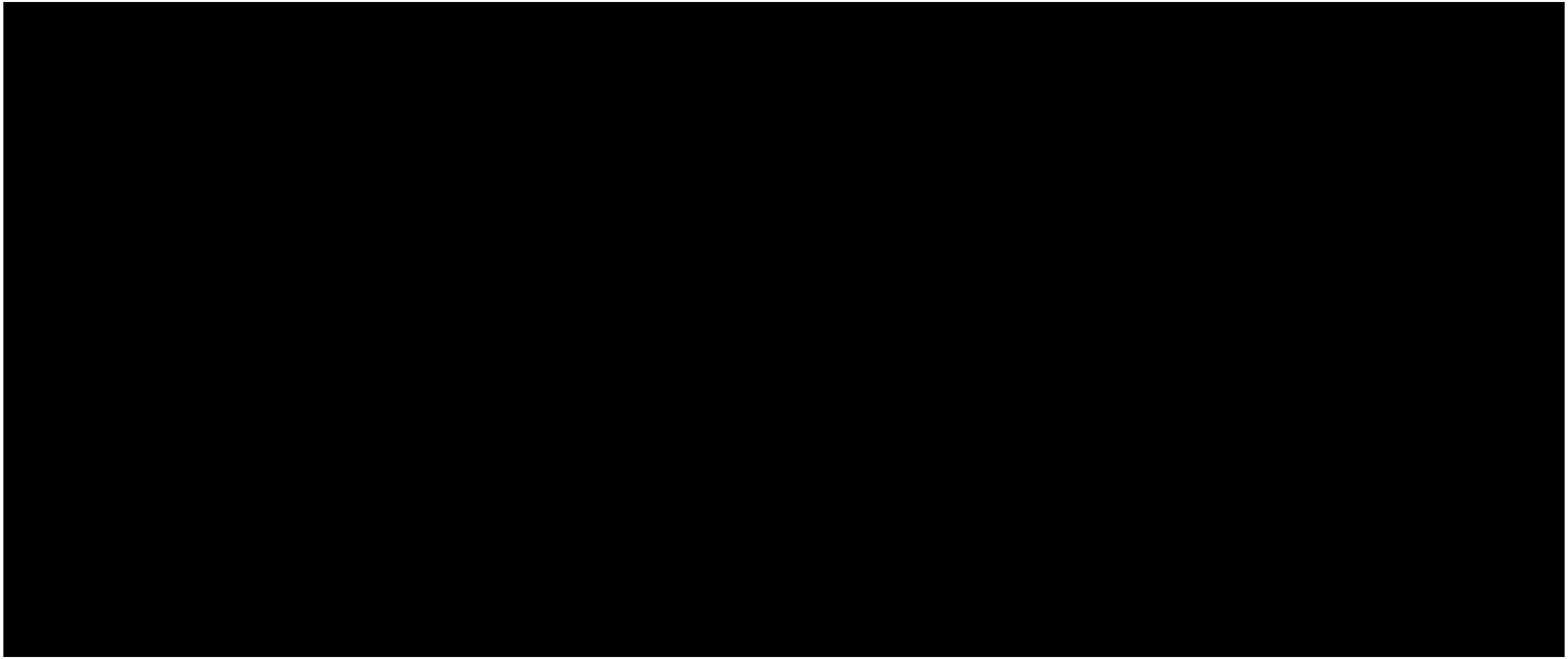
- 1997 to 1998
  - **DreamWorks Animation**

OpenSteer
Steering Behaviors for Autonomous Characters

- OpenSteer is a C++ library to help construct steering behaviors for autonomous characters in games and animation. In addition to the library, OpenSteer provides an OpenGL-based application called OpenSteerDemo which displays predefined demonstrations of steering behaviors. The user can quickly prototype, visualize, annotate and debug new steering behaviors by writing a plug-in for OpenSteerDemo.

- OpenSteer provides a toolkit of steering behaviors, defined in terms of an abstract mobile agent called a "vehicle." Sample code is provided, including a simple vehicle implementation and examples of combining simple steering behaviors to produce more complex behavior. OpenSteer's classes have been designed to flexibly integrate with existing game engines by either layering or inheritance.

张艺谋《长城》特效



Crowd Replication Clip with VFX Breakdown | Created in After Effects

777架无人机点燃长沙七夕之夜（2018）

The End