## horizontale lijn



Technisch ontwerp

08-12-2019

**─**  
  
Bram Nijenkamp  
Buster Bosma  
Felix Beuving  
Jelle van Urk  
Jordy Weening  
René Duivenvoorden

# Inhoudsopgave

[Inhoudsopgave 1](#_Toc36641777)

[Introduction 2](#_Toc36641778)

[1 Steering 3](#_Toc36641779)

[2 Path Planning 4](#_Toc36641780)

[3 Behaviour 5](#_Toc36641781)

[4 Fuzzy Logic 6](#_Toc36641782)

[5. Conclusion 7](#_Toc36641783)

[6 Sources 8](#_Toc36641784)

# Introduction

# 1 Steering

This chapter will explain all the steering behaviour used in our simulator. All behaviours explained in this chapter are based on the code from the book; “Programming game ai by example by Mat Buckland”

## Seek behaviour

This behaviour returns a desired velocity vector based on the target that is given. With this velocity the entity will move towards the target. This behaviour also has the option to stop the entity when in a certain radius of the target.

## Arrive behaviour

## Flocking behaviour

The flocking behaviour creates like the name implies a flocking behaviour where the entities move in unison. Like a flock of birds or a school of fish.   
To create this effect the flocking behaviour uses three behaviours; Seperation, Cohesion and Allignment.

Seperation cause the entity to move away from eachother when within a certain radius.  
Cohesion does the opposit and pulls entities together. And allignement causes entities to move in the same direction.

A problem we encounterd whil making the flocking behaviour. Was that the entities would only go into groups and then not move. This was in issue in the combining of the 3 behaviours. Which we solved by creating the SetMagnitude function in the vector2d class.  
This function normalizes the vector and muliplies it by the magnitude (in this case the max speed).

## Combining behaviours

All moving entities have a list of steering behaviours. When the update function is called all these steering behaviours are calculated and added to the steeringforce. Then the position is updaed using the updatePosition function. This function will also check if the position of the entity is of the screen and will do a wrapAround. Causing the entity to appear on the other side of the screen.

In our simulator the user can activate multiple behaviours using checkboxes. When one of these checkboxes is clicked all entities will get this behaviour in their steeringBehaviour list.   
Causing the entity to follow this behaviour.

# 2 Path Planning

# 3 Behaviour

# 4 Fuzzy Logic

# 

# 5. Conclusion

# 

# 6 Sources

“Programming game ai by example by Mat Buckland”