

Starlings

1.0

Generated by Doxygen 1.8.11

Contents

1	starlings	1
2	Class Index	3
2.1	Class List	3
3	Class Documentation	5
3.1	boid Class Reference	5
3.1.1	Constructor & Destructor Documentation	5
3.1.1.1	boid(float radius=ra, float m=5.0, float min_dis=md)	5
3.1.1.2	boid(position_struct p, velocity_struct v, float radius=ra, float m=5.0, float min_dis=md)	6
3.1.2	Member Function Documentation	6
3.1.2.1	add_position(velocity_struct &p)	6
3.1.2.2	add_velocity(velocity_struct &v)	6
3.1.2.3	alignment(vector< boid > &arr)	6
3.1.2.4	bound_position()	6
3.1.2.5	cohesion(vector< boid > &arr)	7
3.1.2.6	get_mass()	7
3.1.2.7	get_position()	7
3.1.2.8	get_update(vector< boid > &flk, std::vector< obstacle > &obsts, position_struct &goal_pos, velocity_struct &wind)	7
3.1.2.9	get_velocity()	7
3.1.2.10	goal_seeking(position_struct &goal_pos)	7
3.1.2.11	limit_velocity()	7
3.1.2.12	main_rule(vector< boid > &arr, vector< obstacle > &obs)	7

3.1.2.13	print()	8
3.1.2.14	separation(vector< boid > &arr)	8
3.1.2.15	set_position(position_struct &p)	8
3.1.2.16	set_velocity(velocity_struct &v)	8
3.1.3	Member Data Documentation	8
3.1.3.1	mass	8
3.1.3.2	max_position	8
3.1.3.3	max_speed	8
3.1.3.4	minimum_distance	8
3.1.3.5	pos	9
3.1.3.6	radius_of_influence	9
3.1.3.7	vel	9
3.2	flock Class Reference	9
3.2.1	Constructor & Destructor Documentation	10
3.2.1.1	flock()	10
3.2.2	Member Function Documentation	10
3.2.2.1	add_boid()	10
3.2.2.2	add_boid(boid b)	10
3.2.2.3	add_obstacle()	10
3.2.2.4	add_obstacle(obstacle o)	10
3.2.2.5	getBoids()	10
3.2.2.6	getTotalBoids()	10
3.2.2.7	getTotalObstacles()	10
3.2.2.8	render()	11
3.2.2.9	set_goal(position_struct &p)	11
3.2.2.10	spawn(int n=1000)	11
3.2.2.11	update()	11
3.2.2.12	update_goal()	11
3.3	obstacle Class Reference	11
3.3.1	Constructor & Destructor Documentation	12

3.3.1.1	obstacle()	12
3.3.1.2	obstacle(position_struct p, float m=0.0)\	12
3.3.2	Member Function Documentation	12
3.3.2.1	get_mass()	12
3.3.2.2	get_position()	12
3.4	position_struct Struct Reference	12
3.4.1	Detailed Description	13
3.4.2	Constructor & Destructor Documentation	13
3.4.2.1	position_struct()	13
3.4.2.2	position_struct(float a, float b, float c)	13
3.4.3	Member Function Documentation	13
3.4.3.1	add(position_struct &p)	13
3.4.3.2	add(position_struct &v)	13
3.4.3.3	add(float a, float b, float c)	14
3.4.3.4	divide(float a, float b, float c)	14
3.4.3.5	divide(float a)	14
3.4.3.6	multiply(float a)	14
3.4.3.7	normalise()	14
3.4.3.8	set(position_struct p)	14
3.4.3.9	subtract(float a, float b, float c)	15
3.5	velocity_struct Struct Reference	15
3.5.1	Detailed Description	15
3.5.2	Constructor & Destructor Documentation	16
3.5.2.1	velocity_struct()	16
3.5.2.2	velocity_struct(float a, float b, float c)	16
3.5.3	Member Function Documentation	16
3.5.3.1	add(velocity_struct &v)	16
3.5.3.2	add(float a, float b, float c)	16
3.5.3.3	divide(float a, float b, float c)	16
3.5.3.4	divide(float a)	17
3.5.3.5	multiply(float a)	17
3.5.3.6	normalise()	17
3.5.3.7	set(velocity_struct &v)	17
3.5.3.8	subtract(velocity_struct &v)	17
3.5.3.9	subtract(float a, float b, float c)	17

Chapter 1

starlings

Chapter 2

Class Index

2.1 Class List

Here are the classes, structs, unions and interfaces with brief descriptions:

boid	5
flock	9
obstacle	11
position_struct	12
velocity_struct	15

Chapter 3

Class Documentation

3.1 boid Class Reference

Public Member Functions

- [velocity_struct cohesion](#) (vector< [boid](#) > &arr)
- [velocity_struct alignment](#) (vector< [boid](#) > &arr)
- [velocity_struct separation](#) (vector< [boid](#) > &arr)
- [velocity_struct main_rule](#) (vector< [boid](#) > &arr, vector< [obstacle](#) > &obs)
- [velocity_struct bound_position](#) ()
- [velocity_struct goal_seeking](#) ([position_struct](#) &goal_pos)
- void [limit_velocity](#) ()
- [boid](#) (float radius=ra, float m=5.0, float min_dis=md)
- [boid](#) ([position_struct](#) p, [velocity_struct](#) v, float radius=ra, float m=5.0, float min_dis=md)
- [position_struct get_position](#) ()
- [velocity_struct get_velocity](#) ()
- float [get_mass](#) ()
- void [set_position](#) ([position_struct](#) &p)
- void [set_velocity](#) ([velocity_struct](#) &v)
- void [add_velocity](#) ([velocity_struct](#) &v)
- void [add_position](#) ([velocity_struct](#) &p)
- [velocity_struct get_update](#) (vector< [boid](#) > &flk, std::vector< [obstacle](#) > &obsts, [position_struct](#) &goal_pos, [velocity_struct](#) &wind)
- void [print](#) ()

Public Attributes

- [position_struct pos](#)
- [velocity_struct vel](#)
- float [radius_of_influence](#)
- float [mass](#)
- float [minimum_distance](#)
- [position_struct max_position](#)
- float [max_speed](#)

3.1.1 Constructor & Destructor Documentation

3.1.1.1 `boid::boid (float radius = ra, float m = 5 . 0, float min_dis = md)`

Create a new boid

Parameters

<i>radius</i>	radius of influence
<i>m</i>	mass
<i>min_dis</i>	minimum distance

3.1.1.2 `boid::boid (position_struct p, velocity_struct v, float radius = ra, float m = 5.0, float min_dis = md)`

Create a new boid

Parameters

<i>p</i>	initial position
<i>v</i>	initial velocity
<i>radius</i>	radius of influence
<i>m</i>	mass
<i>min_dis</i>	minimum distance

3.1.2 Member Function Documentation

3.1.2.1 `void boid::add_position (velocity_struct & p)`

adds v to current position

Parameters

<i>v</i>	position to be added to current position
----------	--

3.1.2.2 `void boid::add_velocity (velocity_struct & v)`

adds v to current velocity

Parameters

<i>v</i>	velocity to be added to current velocity
----------	--

3.1.2.3 `velocity_struct boid::alignment (vector< boid > & arr)`

returns velocity direction due to alignment property

3.1.2.4 `velocity_struct boid::bound_position ()`

returns velocity direction for bounding position

3.1.2.5 velocity_struct boid::cohesion (vector< boid > & arr)

returns velocity direction due to cohesion

3.1.2.6 float boid::get_mass ()

returns mass of the boid

3.1.2.7 position_struct boid::get_position ()

returns current position of the boid

3.1.2.8 velocity_struct boid::get_update (vector< boid > & flk, std::vector< obstacle > & obs, position_struct & goal_pos, velocity_struct & wind)

calculates the force to be applied to the boid

Parameters

<i>flk</i>	vector containing all boids
<i>obs</i>	vector containing all obstacles
<i>goal_pos</i>	position of goal
<i>wind</i>	velocity due to wind

3.1.2.9 velocity_struct boid::get_velocity ()

returns current velocity of the boid

3.1.2.10 velocity_struct boid::goal_seeking (position_struct & goal_pos)

returns velocity for goal seeking behaviour

3.1.2.11 void boid::limit_velocity ()

limits velocity to max_speed

3.1.2.12 velocity_struct boid::main_rule (vector< boid > & arr, vector< obstacle > & obs)

combines all rules in a single function

3.1.2.13 void boid::print ()

prints current position and velocity of the boid

3.1.2.14 velocity_struct boid::separation (vector< boid > & arr)

returns velocity direction due to separation property

3.1.2.15 void boid::set_position (position_struct & p)

sets current position of boid to p

Parameters

<i>p</i>	new position
----------	--------------

3.1.2.16 void boid::set_velocity (velocity_struct & v)

sets current velocity of boid to p

Parameters

<i>p</i>	new velocity
----------	--------------

3.1.3 Member Data Documentation**3.1.3.1 float boid::mass**

mass of boid

3.1.3.2 position_struct boid::max_position

Boundary of the world (assumed to be a cuboid)

3.1.3.3 float boid::max_speed

Maximum speed a boid can fly with

3.1.3.4 float boid::minimum_distance

minimum distance between two boids

3.1.3.5 position_struct boid::pos

current position of boid

3.1.3.6 float boid::radius_of_influence

distance upto which boid can see

3.1.3.7 velocity_struct boid::vel

current velocity of boid

The documentation for this class was generated from the following files:

- include/boid.h
- src/boid.cpp

3.2 flock Class Reference

Public Member Functions

- flock ()
- void **initialize** ()
- int getTotalBoids ()
- vector< boid > getBoids ()
- void add_boid ()
- void add_boid (boid b)
- void spawn (int n=1000)
- int getTotalObstacles ()
- void set_goal (position_struct &p)
- void update_goal ()
- void add_obstacle ()
- void add_obstacle (obstacle o)
- void update ()
- void render ()
- void **new_render** ()

Public Attributes

- vector< boid > **flk**
- vector< obstacle > **obsts**
- position_struct **goal_pos**
- velocity_struct **wind**

3.2.1 Constructor & Destructor Documentation

3.2.1.1 flock::flock ()

creates a flock

3.2.2 Member Function Documentation

3.2.2.1 void flock::add_boid ()

adds a boid to the flock

3.2.2.2 void flock::add_boid (boid *b*)

adds a boid *b* to the flock

Parameters

<i>b</i>	boid to be added to the flock
----------	-------------------------------

3.2.2.3 void flock::add_obstacle ()

adds an obstacle to the flock

3.2.2.4 void flock::add_obstacle (obstacle *o*)

add an obstacle *o* to the flock

3.2.2.5 vector< boid > flock::getBoids ()

returns a vector containing all boids in the flock

3.2.2.6 int flock::getTotalBoids ()

returns total number of boids in the flock

3.2.2.7 int flock::getTotalObstacles ()

returns total number of obstacles

3.2.2.8 void flock::render ()

renders the flock using OpenGL

3.2.2.9 void flock::set_goal (position_struct & p)

sets the position of goal to p

3.2.2.10 void flock::spawn (int n = 1000)

creates n boids with random initial positions and velocities

Parameters

<i>n</i>	number of boids
----------	-----------------

3.2.2.11 void flock::update ()

updates position and velocities of all the boids in the flock

3.2.2.12 void flock::update_goal ()

updates goal position randomly

The documentation for this class was generated from the following files:

- include/flock.h
- src/flock.cpp

3.3 obstacle Class Reference

Public Member Functions

- [obstacle](#) ()
- [obstacle](#) ([position_struct](#) p, float m=0.0)\
- [position_struct](#) [get_position](#) ()
- float [get_mass](#) ()

Public Attributes

- [position_struct](#) **pos**
- float **mass**

3.3.1 Constructor & Destructor Documentation

3.3.1.1 `obstacle::obstacle ()`

Creates an empty obstacle

3.3.1.2 `obstacle::obstacle (position_struct p, float m = 0.0)`

Creates an obstacle at given position and of given mass

Parameters

<i>p</i>	position of obstacle
<i>m</i>	mass of obstacle

3.3.2 Member Function Documentation

3.3.2.1 `float obstacle::get_mass ()`

Returns mass of the obstacle

3.3.2.2 `position_struct obstacle::get_position ()`

Returns the position of the obstacle

The documentation for this class was generated from the following files:

- include/boid.h
- src/boid.cpp

3.4 position_struct Struct Reference

```
#include <ds.h>
```

Public Member Functions

- [position_struct](#) ()
- [position_struct](#) (float a, float b, float c)
- void [add](#) ([position_struct](#) &p)
- void [add](#) ([position_struct](#) &v)
- void [add](#) (float a, float b, float c)
- void [subtract](#) (float a, float b, float c)
- void [divide](#) (float a, float b, float c)
- void [divide](#) (float a)
- void [multiply](#) (float a)
- void [set](#) ([position_struct](#) p)
- void [normalise](#) ()

Public Attributes

- float **x**
- float **y**
- float **z**

3.4.1 Detailed Description

To store position of the boid

3.4.2 Constructor & Destructor Documentation

3.4.2.1 position_struct::position_struct () [inline]

initializes position to zero

3.4.2.2 position_struct::position_struct (float *a*, float *b*, float *c*) [inline]

initializes position to given value

Parameters

<i>a</i>	x component of position
<i>b</i>	y component of position
<i>c</i>	z component of position

3.4.3 Member Function Documentation

3.4.3.1 void position_struct::add (position_struct & *p*) [inline]

adds position *p* to current position

Parameters

<i>p</i>	position to be added to current position
----------	--

3.4.3.2 void position_struct::add (position_struct & *v*) [inline]

adds velocity *v* to current position

Parameters

<i>v</i>	velocity to be added to current position
----------	--

3.4.3.3 void position_struct::add (float *a*, float *b*, float *c*) [inline]

adds position gicen to current position

Parameters

<i>a</i>	x component of position
<i>b</i>	y component of position
<i>c</i>	z component of position

3.4.3.4 void position_struct::divide (float *a*, float *b*, float *c*) [inline]

divides position by different values in different directions

Parameters

<i>a</i>	x component of position
<i>b</i>	y component of position
<i>c</i>	z component of position

3.4.3.5 void position_struct::divide (float *a*) [inline]

divides position by a uniform value in all different directions

Parameters

<i>a</i>	dividing factor
----------	-----------------

3.4.3.6 void position_struct::multiply (float *a*) [inline]

multiplies position by different values in different directions

Parameters

<i>a</i>	multiplying factors
----------	---------------------

3.4.3.7 void position_struct::normalise () [inline]

converts speed to unity retaining same direction

3.4.3.8 void position_struct::set (position_struct *p*) [inline]

sets position to given value

Parameters

<i>new</i>	position
------------	----------

3.4.3.9 void position_struct::subtract (float *a*, float *b*, float *c*) [inline]

subtract position given to current position

Parameters

<i>a</i>	x component of position
<i>b</i>	y component of position
<i>c</i>	z component of position

The documentation for this struct was generated from the following file:

- include/ds.h

3.5 velocity_struct Struct Reference

```
#include <ds.h>
```

Public Member Functions

- [velocity_struct](#) ()
- [velocity_struct](#) (float *a*, float *b*, float *c*)
- void [add](#) ([velocity_struct](#) &*v*)
- void [add](#) (float *a*, float *b*, float *c*)
- void [subtract](#) ([velocity_struct](#) &*v*)
- void [subtract](#) (float *a*, float *b*, float *c*)
- void [divide](#) (float *a*, float *b*, float *c*)
- void [divide](#) (float *a*)
- void [multiply](#) (float *a*)
- void [set](#) ([velocity_struct](#) &*v*)
- void [set](#) (float *a*, float *b*, float *c*)
- void [normalise](#) ()

Public Attributes

- float **x**
- float **y**
- float **z**

3.5.1 Detailed Description

To store velocity of the boid

3.5.2 Constructor & Destructor Documentation

3.5.2.1 `velocity_struct::velocity_struct ()` `[inline]`

initializes velocity to zero

3.5.2.2 `velocity_struct::velocity_struct (float a, float b, float c)` `[inline]`

initializes velocity to given value

Parameters

<i>a</i>	x component of velocity
<i>b</i>	y component of velocity
<i>c</i>	z component of velocity

3.5.3 Member Function Documentation

3.5.3.1 `void velocity_struct::add (velocity_struct & v)` `[inline]`

adds velocity *v* to current velocity

Parameters

<i>v</i>	velocity to be added to current velocity
----------	--

3.5.3.2 `void velocity_struct::add (float a, float b, float c)` `[inline]`

adds velocity gicen to current velocity

Parameters

<i>a</i>	x component of velocity
<i>b</i>	y component of velocity
<i>c</i>	z component of velocity

3.5.3.3 `void velocity_struct::divide (float a, float b, float c)` `[inline]`

divides velocity by different values in different directions

Parameters

<i>a</i>	x component of velocity
<i>b</i>	y component of velocity
<i>c</i>	z component of velocity

3.5.3.4 void velocity_struct::divide (float *a*) [inline]

divides velocity by a uniform value in all different directions

Parameters

<i>a</i>	dividing factor
----------	-----------------

3.5.3.5 void velocity_struct::multiply (float *a*) [inline]

multiplies velocity by different values in different directions

Parameters

<i>a</i>	multiplying factors
----------	---------------------

3.5.3.6 void velocity_struct::normalise () [inline]

converts distance to unity retaining same direction

3.5.3.7 void velocity_struct::set (velocity_struct & *v*) [inline]

sets position to given value

Parameters

<i>v</i>	new position
----------	--------------

3.5.3.8 void velocity_struct::subtract (velocity_struct & *v*) [inline]

subtracts velocity *v* to current velocity

Parameters

<i>v</i>	velocity to be subtracted to current velocity
----------	---

3.5.3.9 void velocity_struct::subtract (float *a*, float *b*, float *c*) [inline]

subtract velocity given to current velocity

Parameters

<i>a</i>	x component of velocity
----------	-------------------------

Parameters

<i>b</i>	y component of velocity
<i>c</i>	z component of velocity

The documentation for this struct was generated from the following file:

- include/ds.h

Index

- add
 - position_struct, [13](#), [14](#)
 - velocity_struct, [16](#)
- add_boid
 - flock, [10](#)
- add_obstacle
 - flock, [10](#)
- add_position
 - boid, [6](#)
- add_velocity
 - boid, [6](#)
- alignment
 - boid, [6](#)
- boid, [5](#)
 - add_position, [6](#)
 - add_velocity, [6](#)
 - alignment, [6](#)
 - boid, [5](#), [6](#)
 - bound_position, [6](#)
 - cohesion, [6](#)
 - get_mass, [7](#)
 - get_position, [7](#)
 - get_update, [7](#)
 - get_velocity, [7](#)
 - goal_seeking, [7](#)
 - limit_velocity, [7](#)
 - main_rule, [7](#)
 - mass, [8](#)
 - max_position, [8](#)
 - max_speed, [8](#)
 - minimum_distance, [8](#)
 - pos, [8](#)
 - print, [7](#)
 - radius_of_influence, [9](#)
 - separation, [8](#)
 - set_position, [8](#)
 - set_velocity, [8](#)
 - vel, [9](#)
- bound_position
 - boid, [6](#)
- cohesion
 - boid, [6](#)
- divide
 - position_struct, [14](#)
 - velocity_struct, [16](#), [17](#)
- flock, [9](#)
 - add_boid, [10](#)
 - add_obstacle, [10](#)
 - flock, [10](#)
 - getBoids, [10](#)
 - getTotalBoids, [10](#)
 - getTotalObstacles, [10](#)
 - render, [10](#)
 - set_goal, [11](#)
 - spawn, [11](#)
 - update, [11](#)
 - update_goal, [11](#)
- get_mass
 - boid, [7](#)
 - obstacle, [12](#)
- get_position
 - boid, [7](#)
 - obstacle, [12](#)
- get_update
 - boid, [7](#)
- get_velocity
 - boid, [7](#)
- getBoids
 - flock, [10](#)
- getTotalBoids
 - flock, [10](#)
- getTotalObstacles
 - flock, [10](#)
- goal_seeking
 - boid, [7](#)
- limit_velocity
 - boid, [7](#)
- main_rule
 - boid, [7](#)
- mass
 - boid, [8](#)
- max_position
 - boid, [8](#)
- max_speed
 - boid, [8](#)
- minimum_distance
 - boid, [8](#)
- multiply
 - position_struct, [14](#)
 - velocity_struct, [17](#)
- normalise
 - position_struct, [14](#)

- velocity_struct, [17](#)
- obstacle, [11](#)
 - get_mass, [12](#)
 - get_position, [12](#)
 - obstacle, [12](#)
- pos
 - boid, [8](#)
- position_struct, [12](#)
 - add, [13](#), [14](#)
 - divide, [14](#)
 - multiply, [14](#)
 - normalise, [14](#)
 - position_struct, [13](#)
 - set, [14](#)
 - subtract, [15](#)
- print
 - boid, [7](#)
- radius_of_influence
 - boid, [9](#)
- render
 - flock, [10](#)
- separation
 - boid, [8](#)
- set
 - position_struct, [14](#)
 - velocity_struct, [17](#)
- set_goal
 - flock, [11](#)
- set_position
 - boid, [8](#)
- set_velocity
 - boid, [8](#)
- spawn
 - flock, [11](#)
- subtract
 - position_struct, [15](#)
 - velocity_struct, [17](#)
- update
 - flock, [11](#)
- update_goal
 - flock, [11](#)
- vel
 - boid, [9](#)
- velocity_struct, [15](#)
 - add, [16](#)
 - divide, [16](#), [17](#)
 - multiply, [17](#)
 - normalise, [17](#)
 - set, [17](#)
 - subtract, [17](#)
 - velocity_struct, [16](#)