

## Autonomous Steering Agents

Generated by Doxygen 1.8.17



|  |           |
|--|-----------|
| <b>1 Intent</b>                              | <b>1</b>  |
| 1.1 Dependencies                             | 1         |
| 1.2 Resources                                | 1         |
| 1.3 Links                                    | 1         |
| <b>2 Todo List</b>                           | <b>3</b>  |
| <b>3 Hierarchical Index</b>                  | <b>5</b>  |
| 3.1 Class Hierarchy                          | 5         |
| <b>4 Class Index</b>                         | <b>7</b>  |
| 4.1 Class List                               | 7         |
| <b>5 File Index</b>                          | <b>9</b>  |
| 5.1 File List                                | 9         |
| <b>6 Class Documentation</b>                 | <b>13</b> |
| 6.1 agent Class Reference                    | 13        |
| 6.1.1 Detailed Description                   | 14        |
| 6.1.2 Constructor & Destructor Documentation | 14        |
| 6.1.2.1 agent() [1/2]                        | 14        |
| 6.1.2.2 agent() [2/2]                        | 14        |
| 6.1.2.3 ~agent()                             | 15        |
| 6.1.3 Member Function Documentation          | 15        |
| 6.1.3.1 draw()                               | 15        |
| 6.1.3.2 getMass()                            | 16        |
| 6.1.3.3 getName()                            | 16        |
| 6.1.3.4 getTarget()                          | 16        |
| 6.1.3.5 getVelocity()                        | 16        |
| 6.1.3.6 setFeatures()                        | 16        |
| 6.1.3.7 setMass()                            | 17        |
| 6.1.3.8 setName()                            | 17        |
| 6.1.3.9 setTarget()                          | 18        |
| 6.1.3.10 setVelocity()                       | 18        |
| 6.1.3.11 updatePosition()                    | 18        |
| 6.1.4 Member Data Documentation              | 19        |
| 6.1.4.1 acceleration                         | 19        |
| 6.1.4.2 arrive                               | 19        |
| 6.1.4.3 desiredVelocity                      | 19        |
| 6.1.4.4 force                                | 20        |
| 6.1.4.5 id                                   | 20        |
| 6.1.4.6 maxForce                             | 20        |
| 6.1.4.7 maxSpeed                             | 20        |
| 6.1.4.8 position                             | 20        |

|  |    |
|--|----|
| 6.1.4.9 r                                    | 21 |
| 6.1.4.10 steering                            | 21 |
| 6.2 color Class Reference                    | 21 |
| 6.2.1 Detailed Description                   | 22 |
| 6.2.2 Constructor & Destructor Documentation | 22 |
| 6.2.2.1 color() [1/2]                        | 22 |
| 6.2.2.2 color() [2/2]                        | 22 |
| 6.2.3 Member Function Documentation          | 23 |
| 6.2.3.1 getColor()                           | 23 |
| 6.2.4 Member Data Documentation              | 23 |
| 6.2.4.1 B                                    | 23 |
| 6.2.4.2 G                                    | 23 |
| 6.2.4.3 R                                    | 24 |
| 6.3 entity Class Reference                   | 24 |
| 6.3.1 Detailed Description                   | 24 |
| 6.3.2 Constructor & Destructor Documentation | 24 |
| 6.3.2.1 entity()                             | 25 |
| 6.3.3 Member Function Documentation          | 25 |
| 6.3.3.1 draw()                               | 25 |
| 6.3.3.2 getColor()                           | 25 |
| 6.3.3.3 getId()                              | 25 |
| 6.3.3.4 getName()                            | 26 |
| 6.3.3.5 setColor()                           | 26 |
| 6.3.3.6 setId()                              | 26 |
| 6.3.3.7 setName()                            | 26 |
| 6.4 evade Class Reference                    | 27 |
| 6.4.1 Detailed Description                   | 27 |
| 6.4.2 Constructor & Destructor Documentation | 27 |
| 6.4.2.1 evade()                              | 27 |
| 6.4.3 Member Function Documentation          | 28 |
| 6.4.3.1 loop()                               | 28 |
| 6.5 flee Class Reference                     | 28 |
| 6.5.1 Detailed Description                   | 29 |
| 6.5.2 Constructor & Destructor Documentation | 29 |
| 6.5.2.1 flee()                               | 29 |
| 6.5.3 Member Function Documentation          | 29 |
| 6.5.3.1 loop()                               | 29 |
| 6.6 flock Class Reference                    | 30 |
| 6.6.1 Detailed Description                   | 30 |
| 6.6.2 Constructor & Destructor Documentation | 30 |
| 6.6.2.1 flock()                              | 30 |
| 6.6.3 Member Function Documentation          | 30 |

|   |    |
|---|----|
| 6.6.3.1 loop()                                | 31 |
| 6.7 flowField Class Reference                 | 31 |
| 6.7.1 Detailed Description                    | 31 |
| 6.7.2 Constructor & Destructor Documentation  | 31 |
| 6.7.2.1 flowField() [1/2]                     | 32 |
| 6.7.2.2 flowField() [2/2]                     | 32 |
| 6.7.3 Member Function Documentation           | 32 |
| 6.7.3.1 getField()                            | 32 |
| 6.8 graphics Class Reference                  | 33 |
| 6.8.1 Detailed Description                    | 34 |
| 6.8.2 Member Function Documentation           | 34 |
| 6.8.2.1 drawAgent()                           | 34 |
| 6.8.2.2 drawCircle()                          | 35 |
| 6.8.2.3 drawLine()                            | 35 |
| 6.8.2.4 drawPath()                            | 36 |
| 6.8.2.5 drawPoint()                           | 36 |
| 6.8.2.6 drawText()                            | 36 |
| 6.8.2.7 forceInScreen()                       | 37 |
| 6.8.2.8 getMousePosition()                    | 37 |
| 6.8.2.9 handleKeypress()                      | 38 |
| 6.8.2.10 handleResize()                       | 38 |
| 6.8.2.11 initGraphics()                       | 38 |
| 6.8.2.12 mouseButton()                        | 40 |
| 6.8.2.13 mouseMove()                          | 40 |
| 6.8.2.14 refreshScene()                       | 41 |
| 6.8.2.15 timerEvent()                         | 41 |
| 6.8.3 Member Data Documentation               | 41 |
| 6.8.3.1 target_x                              | 42 |
| 6.8.3.2 target_y                              | 42 |
| 6.9 leaderFollower Class Reference            | 42 |
| 6.9.1 Detailed Description                    | 43 |
| 6.9.2 Constructor & Destructor Documentation  | 43 |
| 6.9.2.1 leaderFollower()                      | 43 |
| 6.9.3 Member Function Documentation           | 43 |
| 6.9.3.1 loop()                                | 43 |
| 6.9.4 Member Data Documentation               | 44 |
| 6.9.4.1 leaderPosition                        | 44 |
| 6.9.4.2 leaderVelocity                        | 44 |
| 6.10 mouseFollower Class Reference            | 45 |
| 6.10.1 Detailed Description                   | 45 |
| 6.10.2 Constructor & Destructor Documentation | 45 |
| 6.10.2.1 mouseFollower()                      | 45 |

---

|   |    |
|---|----|
| 6.10.3 Member Function Documentation          | 45 |
| 6.10.3.1 loop()                               | 46 |
| 6.11 obstacle Class Reference                 | 46 |
| 6.11.1 Detailed Description                   | 46 |
| 6.11.2 Constructor & Destructor Documentation | 47 |
| 6.11.2.1 obstacle() [1/2]                     | 47 |
| 6.11.2.2 obstacle() [2/2]                     | 47 |
| 6.11.3 Member Function Documentation          | 47 |
| 6.11.3.1 draw()                               | 48 |
| 6.11.3.2 getCenter()                          | 48 |
| 6.11.3.3 getRadius()                          | 48 |
| 6.11.3.4 setCenter()                          | 48 |
| 6.11.3.5 setRadius()                          | 49 |
| 6.12 obstacleAvoidance Class Reference        | 49 |
| 6.12.1 Detailed Description                   | 49 |
| 6.12.2 Constructor & Destructor Documentation | 49 |
| 6.12.2.1 obstacleAvoidance()                  | 50 |
| 6.12.3 Member Function Documentation          | 50 |
| 6.12.3.1 createObstacle()                     | 50 |
| 6.12.3.2 loop()                               | 50 |
| 6.12.4 Member Data Documentation              | 51 |
| 6.12.4.1 obstacles                            | 51 |
| 6.13 path Class Reference                     | 51 |
| 6.13.1 Detailed Description                   | 52 |
| 6.13.2 Constructor & Destructor Documentation | 52 |
| 6.13.2.1 path() [1/2]                         | 52 |
| 6.13.2.2 path() [2/2]                         | 52 |
| 6.13.3 Member Function Documentation          | 53 |
| 6.13.3.1 addPoint()                           | 53 |
| 6.13.3.2 draw()                               | 53 |
| 6.13.3.3 getPathWidth()                       | 54 |
| 6.13.3.4 setPathWidth()                       | 54 |
| 6.13.4 Member Data Documentation              | 54 |
| 6.13.4.1 points                               | 54 |
| 6.14 pathFollower Class Reference             | 54 |
| 6.14.1 Detailed Description                   | 55 |
| 6.14.2 Constructor & Destructor Documentation | 55 |
| 6.14.2.1 pathFollower()                       | 55 |
| 6.14.3 Member Function Documentation          | 55 |
| 6.14.3.1 createPath()                         | 55 |
| 6.14.3.2 loop()                               | 56 |
| 6.14.4 Member Data Documentation              | 56 |

---

|   |    |
|---|----|
| 6.14.4.1 myPath . . . . .                               | 56 |
| 6.15 point Class Reference . . . . .                    | 57 |
| 6.15.1 Detailed Description . . . . .                   | 57 |
| 6.15.2 Constructor & Destructor Documentation . . . . . | 57 |
| 6.15.2.1 point() [1/2] . . . . .                        | 58 |
| 6.15.2.2 point() [2/2] . . . . .                        | 58 |
| 6.15.3 Member Function Documentation . . . . .          | 58 |
| 6.15.3.1 div() . . . . .                                | 58 |
| 6.15.3.2 getNormalPoint() . . . . .                     | 59 |
| 6.15.3.3 mul() . . . . .                                | 59 |
| 6.15.3.4 operator+() [1/2] . . . . .                    | 60 |
| 6.15.3.5 operator+() [2/2] . . . . .                    | 60 |
| 6.15.3.6 operator-() . . . . .                          | 61 |
| 6.15.3.7 operator==() . . . . .                         | 61 |
| 6.15.3.8 print() . . . . .                              | 61 |
| 6.15.4 Member Data Documentation . . . . .              | 62 |
| 6.15.4.1 x . . . . .                                    | 62 |
| 6.15.4.2 y . . . . .                                    | 62 |
| 6.16 prison Class Reference . . . . .                   | 62 |
| 6.16.1 Detailed Description . . . . .                   | 63 |
| 6.16.2 Constructor & Destructor Documentation . . . . . | 63 |
| 6.16.2.1 prison() . . . . .                             | 63 |
| 6.16.3 Member Function Documentation . . . . .          | 63 |
| 6.16.3.1 loop() . . . . .                               | 63 |
| 6.17 pursuit Class Reference . . . . .                  | 64 |
| 6.17.1 Detailed Description . . . . .                   | 64 |
| 6.17.2 Constructor & Destructor Documentation . . . . . | 64 |
| 6.17.2.1 pursuit() . . . . .                            | 64 |
| 6.17.3 Member Function Documentation . . . . .          | 64 |
| 6.17.3.1 loop() . . . . .                               | 65 |
| 6.18 pvector Class Reference . . . . .                  | 65 |
| 6.18.1 Detailed Description . . . . .                   | 66 |
| 6.18.2 Constructor & Destructor Documentation . . . . . | 66 |
| 6.18.2.1 pvector() [1/2] . . . . .                      | 66 |
| 6.18.2.2 pvector() [2/2] . . . . .                      | 67 |
| 6.18.3 Member Function Documentation . . . . .          | 67 |
| 6.18.3.1 add() . . . . .                                | 67 |
| 6.18.3.2 angleBetween() . . . . .                       | 67 |
| 6.18.3.3 div() . . . . .                                | 68 |
| 6.18.3.4 dotProduct() . . . . .                         | 68 |
| 6.18.3.5 getAngle() . . . . .                           | 69 |
| 6.18.3.6 limit() . . . . .                              | 69 |

|   |    |
|---|----|
| 6.18.3.7 magnitude()                          | 69 |
| 6.18.3.8 mul()                                | 70 |
| 6.18.3.9 normalize()                          | 70 |
| 6.18.3.10 operator+() [1/2]                   | 70 |
| 6.18.3.11 operator+() [2/2]                   | 71 |
| 6.18.3.12 operator+=()                        | 71 |
| 6.18.3.13 operator-() [1/2]                   | 72 |
| 6.18.3.14 operator-() [2/2]                   | 72 |
| 6.18.3.15 operator==()                        | 73 |
| 6.18.3.16 print()                             | 73 |
| 6.18.4 Member Data Documentation              | 73 |
| 6.18.4.1 x                                    | 73 |
| 6.18.4.2 y                                    | 74 |
| 6.19 random Class Reference                   | 74 |
| 6.19.1 Detailed Description                   | 74 |
| 6.19.2 Member Function Documentation          | 74 |
| 6.19.2.1 createRandomArray()                  | 74 |
| 6.20 scenario Class Reference                 | 75 |
| 6.20.1 Detailed Description                   | 76 |
| 6.20.2 Constructor & Destructor Documentation | 76 |
| 6.20.2.1 scenario()                           | 76 |
| 6.20.3 Member Function Documentation          | 76 |
| 6.20.3.1 createRandomAgents()                 | 76 |
| 6.20.3.2 createStaticAgents()                 | 77 |
| 6.20.3.3 createTroop()                        | 77 |
| 6.20.3.4 initGL()                             | 78 |
| 6.20.3.5 refresh()                            | 78 |
| 6.20.4 Member Data Documentation              | 79 |
| 6.20.4.1 agents                               | 79 |
| 6.20.4.2 behavior                             | 79 |
| 6.20.4.3 callback                             | 79 |
| 6.20.4.4 name                                 | 79 |
| 6.20.4.5 view                                 | 80 |
| 6.21 steeringBehavior Class Reference         | 80 |
| 6.21.1 Detailed Description                   | 81 |
| 6.21.2 Member Function Documentation          | 81 |
| 6.21.2.1 align()                              | 81 |
| 6.21.2.2 avoid()                              | 81 |
| 6.21.2.3 cohesion()                           | 82 |
| 6.21.2.4 evade()                              | 83 |
| 6.21.2.5 flee()                               | 84 |
| 6.21.2.6 inFlowField()                        | 84 |



|   |           |
|---|-----------|
| 6.21.2.7 pursuit()  | 85        |
| 6.21.2.8 seek()   | 85        |
| 6.21.2.9 separation()   | 86        |
| 6.21.2.10 stayInArea()  | 87        |
| 6.21.2.11 stayInPath()  | 87        |
| 6.21.2.12 wander()  | 88        |
| 6.22 wander Class Reference   | 89        |
| 6.22.1 Detailed Description   | 89        |
| 6.22.2 Constructor & Destructor Documentation                                       | 89        |
| 6.22.2.1 wander()   | 89        |
| 6.22.3 Member Function Documentation  | 89        |
| 6.22.3.1 loop()   | 90        |
| 6.23 windy Class Reference  | 90        |
| 6.23.1 Detailed Description   | 90        |
| 6.23.2 Constructor & Destructor Documentation                                       | 91        |
| 6.23.2.1 windy()  | 91        |
| 6.23.3 Member Function Documentation  | 91        |
| 6.23.3.1 loop()   | 91        |
| 6.23.4 Member Data Documentation  | 91        |
| 6.23.4.1 flow   | 91        |
| <b>7 File Documentation</b>   | <b>93</b> |
| 7.1 /home/user/Desktop/mmm/autonomousSteeringAgents/agent.d File Reference          | 93        |
| 7.2 /home/user/Desktop/mmm/autonomousSteeringAgents/color.d File Reference          | 93        |
| 7.3 /home/user/Desktop/mmm/autonomousSteeringAgents/entity.d File Reference         | 93        |
| 7.4 /home/user/Desktop/mmm/autonomousSteeringAgents/evade.d File Reference          | 93        |
| 7.5 /home/user/Desktop/mmm/autonomousSteeringAgents/flee.d File Reference           | 93        |
| 7.6 /home/user/Desktop/mmm/autonomousSteeringAgents/flock.d File Reference          | 93        |
| 7.7 /home/user/Desktop/mmm/autonomousSteeringAgents/flowField.d File Reference      | 93        |
| 7.8 /home/user/Desktop/mmm/autonomousSteeringAgents/graphics.d File Reference       | 93        |
| 7.9 /home/user/Desktop/mmm/autonomousSteeringAgents/include/agent.h File Reference  | 93        |
| 7.9.1 Detailed Description  | 94        |
| 7.10 /home/user/Desktop/mmm/autonomousSteeringAgents/include/color.h File Reference | 95        |
| 7.10.1 Detailed Description   | 96        |
| 7.10.2 Macro Definition Documentation   | 96        |
| 7.10.2.1 BLACK  | 96        |
| 7.10.2.2 BLUE   | 96        |
| 7.10.2.3 CYAN   | 96        |
| 7.10.2.4 GREEN  | 96        |
| 7.10.2.5 MAGENDA  | 97        |
| 7.10.2.6 RED  | 97        |
| 7.10.2.7 WHITE  | 97        |

|   |     |
|---|-----|
| 7.10.2.8 YELLOW . . . . .   | 97  |
| 7.11 /home/user/Desktop/mmm/autonomousSteeringAgents/include/entity.h File Reference . . . . .            | 97  |
| 7.12 /home/user/Desktop/mmm/autonomousSteeringAgents/include/evade.h File Reference . . . . .             | 98  |
| 7.12.1 Detailed Description . . . . .   | 99  |
| 7.13 /home/user/Desktop/mmm/autonomousSteeringAgents/include/flee.h File Reference . . . . .              | 99  |
| 7.13.1 Detailed Description . . . . .   | 100 |
| 7.14 /home/user/Desktop/mmm/autonomousSteeringAgents/include/flock.h File Reference . . . . .             | 101 |
| 7.14.1 Detailed Description . . . . .   | 102 |
| 7.15 /home/user/Desktop/mmm/autonomousSteeringAgents/include/flowField.h File Reference . . . . .         | 102 |
| 7.15.1 Detailed Description . . . . .   | 103 |
| 7.15.2 Macro Definition Documentation . . . . .   | 104 |
| 7.15.2.1 FIELD_HEIGHT . . . . .   | 104 |
| 7.15.2.2 FIELD_WIDTH . . . . .  | 104 |
| 7.15.2.3 GRAVITY . . . . .  | 104 |
| 7.15.2.4 WIND_WEST . . . . .  | 104 |
| 7.16 /home/user/Desktop/mmm/autonomousSteeringAgents/include/graphics.h File Reference . . . . .          | 105 |
| 7.16.1 Detailed Description . . . . .   | 106 |
| 7.16.2 Macro Definition Documentation . . . . .   | 106 |
| 7.16.2.1 ESC . . . . .  | 106 |
| 7.16.2.2 HEIGHT . . . . .   | 106 |
| 7.16.2.3 PI . . . . .   | 106 |
| 7.16.2.4 WIDTH . . . . .  | 106 |
| 7.17 /home/user/Desktop/mmm/autonomousSteeringAgents/include/leaderFollower.h File Reference . . . . .    | 107 |
| 7.17.1 Detailed Description . . . . .   | 108 |
| 7.18 /home/user/Desktop/mmm/autonomousSteeringAgents/include/mouseFollower.h File Reference . . . . .     | 108 |
| 7.18.1 Detailed Description . . . . .   | 109 |
| 7.19 /home/user/Desktop/mmm/autonomousSteeringAgents/include/obstacle.h File Reference . . . . .          | 109 |
| 7.19.1 Detailed Description . . . . .   | 110 |
| 7.20 /home/user/Desktop/mmm/autonomousSteeringAgents/include/obstacleAvoidance.h File Reference . . . . . | 111 |
| 7.20.1 Detailed Description . . . . .   | 112 |
| 7.21 /home/user/Desktop/mmm/autonomousSteeringAgents/include/path.h File Reference . . . . .              | 112 |
| 7.21.1 Detailed Description . . . . .   | 113 |
| 7.22 /home/user/Desktop/mmm/autonomousSteeringAgents/include/pathFollower.h File Reference . . . . .      | 113 |
| 7.22.1 Detailed Description . . . . .   | 114 |
| 7.23 /home/user/Desktop/mmm/autonomousSteeringAgents/include/point.h File Reference . . . . .             | 115 |
| 7.23.1 Detailed Description . . . . .   | 116 |
| 7.24 /home/user/Desktop/mmm/autonomousSteeringAgents/include/prison.h File Reference . . . . .            | 116 |
| 7.24.1 Detailed Description . . . . .   | 117 |
| 7.25 /home/user/Desktop/mmm/autonomousSteeringAgents/include/pursuit.h File Reference . . . . .           | 117 |
| 7.25.1 Detailed Description . . . . .   | 118 |
| 7.26 /home/user/Desktop/mmm/autonomousSteeringAgents/include/pvector.h File Reference . . . . .           | 119 |
| 7.26.1 Detailed Description . . . . .   | 120 |

|  |     |
|--|-----|
| 7.26.2 Macro Definition Documentation . . . . .  | 120 |
| 7.26.2.1 PI . . . . .  | 120 |
| 7.27 /home/user/Desktop/mmm/autonomousSteeringAgents/include/random.h File Reference . . . . .           | 120 |
| 7.27.1 Detailed Description . . . . .  | 121 |
| 7.28 /home/user/Desktop/mmm/autonomousSteeringAgents/include/scenario.h File Reference . . . . .         | 121 |
| 7.28.1 Detailed Description . . . . .  | 122 |
| 7.28.2 Enumeration Type Documentation . . . . .  | 122 |
| 7.28.2.1 types . . . . .   | 122 |
| 7.29 /home/user/Desktop/mmm/autonomousSteeringAgents/include/steeringBehavior.h File Reference . . . . . | 122 |
| 7.29.1 Detailed Description . . . . .  | 124 |
| 7.29.2 Macro Definition Documentation . . . . .  | 124 |
| 7.29.2.1 AVOID_OBSTACLE . . . . .  | 124 |
| 7.29.2.2 CIRCLE_DISTANCE . . . . .   | 124 |
| 7.29.2.3 CIRCLE_RADIUS . . . . .   | 124 |
| 7.29.2.4 EVADE . . . . .   | 124 |
| 7.29.2.5 FLEE . . . . .  | 125 |
| 7.29.2.6 FLOCK . . . . .   | 125 |
| 7.29.2.7 FOLLOW_MOUSE . . . . .  | 125 |
| 7.29.2.8 IN_FLOW_FIELD . . . . .   | 125 |
| 7.29.2.9 LEADER_FOLLOWER . . . . .   | 125 |
| 7.29.2.10 PURSUIT . . . . .  | 125 |
| 7.29.2.11 STAY_IN_FIELD . . . . .  | 126 |
| 7.29.2.12 STAY_IN_PATH . . . . .   | 126 |
| 7.29.2.13 WANDER . . . . .   | 126 |
| 7.30 /home/user/Desktop/mmm/autonomousSteeringAgents/include/wander.h File Reference . . . . .           | 126 |
| 7.30.1 Detailed Description . . . . .  | 127 |
| 7.31 /home/user/Desktop/mmm/autonomousSteeringAgents/include/windy.h File Reference . . . . .            | 128 |
| 7.31.1 Detailed Description . . . . .  | 129 |
| 7.32 /home/user/Desktop/mmm/autonomousSteeringAgents/leaderFollower.d File Reference . . . . .           | 129 |
| 7.33 /home/user/Desktop/mmm/autonomousSteeringAgents/main.cpp File Reference . . . . .                   | 129 |
| 7.33.1 Detailed Description . . . . .  | 130 |
| 7.33.2 Function Documentation . . . . .  | 130 |
| 7.33.2.1 main() . . . . .  | 131 |
| 7.33.2.2 menu() . . . . .  | 131 |
| 7.33.3 Variable Documentation . . . . .  | 132 |
| 7.33.3.1 mode . . . . .  | 132 |
| 7.34 /home/user/Desktop/mmm/autonomousSteeringAgents/main.d File Reference . . . . .                     | 134 |
| 7.35 /home/user/Desktop/mmm/autonomousSteeringAgents/mouseFollower.d File Reference . . . . .            | 134 |
| 7.36 /home/user/Desktop/mmm/autonomousSteeringAgents/obstacle.d File Reference . . . . .                 | 134 |
| 7.37 /home/user/Desktop/mmm/autonomousSteeringAgents/obstacleAvoidance.d File Reference . . . . .        | 134 |
| 7.38 /home/user/Desktop/mmm/autonomousSteeringAgents/path.d File Reference . . . . .                     | 134 |
| 7.39 /home/user/Desktop/mmm/autonomousSteeringAgents/pathFollower.d File Reference . . . . .             | 134 |

|   |     |
|---|-----|
| 7.40 /home/user/Desktop/mmm/autonomousSteeringAgents/point.d File Reference . . . . .                   | 134 |
| 7.41 /home/user/Desktop/mmm/autonomousSteeringAgents/prison.d File Reference . . . . .                  | 134 |
| 7.42 /home/user/Desktop/mmm/autonomousSteeringAgents/pursuit.d File Reference . . . . .                 | 134 |
| 7.43 /home/user/Desktop/mmm/autonomousSteeringAgents/pvector.d File Reference . . . . .                 | 134 |
| 7.44 /home/user/Desktop/mmm/autonomousSteeringAgents/random.d File Reference . . . . .                  | 134 |
| 7.45 /home/user/Desktop/mmm/autonomousSteeringAgents/README.md File Reference . . . . .                 | 134 |
| 7.46 /home/user/Desktop/mmm/autonomousSteeringAgents/scenario.d File Reference . . . . .                | 134 |
| 7.47 /home/user/Desktop/mmm/autonomousSteeringAgents/src/agent.cpp File Reference . . . . .             | 134 |
| 7.47.1 Detailed Description . . . . .   | 135 |
| 7.48 /home/user/Desktop/mmm/autonomousSteeringAgents/src/color.cpp File Reference . . . . .             | 135 |
| 7.48.1 Detailed Description . . . . .   | 136 |
| 7.49 /home/user/Desktop/mmm/autonomousSteeringAgents/src/entity.cpp File Reference . . . . .            | 136 |
| 7.49.1 Detailed Description . . . . .   | 137 |
| 7.50 /home/user/Desktop/mmm/autonomousSteeringAgents/src/evade.cpp File Reference . . . . .             | 137 |
| 7.50.1 Detailed Description . . . . .   | 138 |
| 7.51 /home/user/Desktop/mmm/autonomousSteeringAgents/src/flee.cpp File Reference . . . . .              | 138 |
| 7.51.1 Detailed Description . . . . .   | 139 |
| 7.52 /home/user/Desktop/mmm/autonomousSteeringAgents/src/flock.cpp File Reference . . . . .             | 139 |
| 7.52.1 Detailed Description . . . . .   | 140 |
| 7.53 /home/user/Desktop/mmm/autonomousSteeringAgents/src/flowField.cpp File Reference . . . . .         | 140 |
| 7.53.1 Detailed Description . . . . .   | 141 |
| 7.54 /home/user/Desktop/mmm/autonomousSteeringAgents/src/graphics.cpp File Reference . . . . .          | 141 |
| 7.54.1 Detailed Description . . . . .   | 142 |
| 7.55 /home/user/Desktop/mmm/autonomousSteeringAgents/src/leaderFollower.cpp File Reference . . . . .    | 142 |
| 7.55.1 Detailed Description . . . . .   | 143 |
| 7.56 /home/user/Desktop/mmm/autonomousSteeringAgents/src/mouseFollower.cpp File Reference . . . . .     | 143 |
| 7.56.1 Detailed Description . . . . .   | 144 |
| 7.57 /home/user/Desktop/mmm/autonomousSteeringAgents/src/obstacle.cpp File Reference . . . . .          | 144 |
| 7.57.1 Detailed Description . . . . .   | 145 |
| 7.58 /home/user/Desktop/mmm/autonomousSteeringAgents/src/obstacleAvoidance.cpp File Reference . . . . . | 145 |
| 7.58.1 Detailed Description . . . . .   | 146 |
| 7.59 /home/user/Desktop/mmm/autonomousSteeringAgents/src/path.cpp File Reference . . . . .              | 146 |
| 7.59.1 Detailed Description . . . . .   | 147 |
| 7.60 /home/user/Desktop/mmm/autonomousSteeringAgents/src/pathFollower.cpp File Reference . . . . .      | 147 |
| 7.60.1 Detailed Description . . . . .   | 148 |
| 7.61 /home/user/Desktop/mmm/autonomousSteeringAgents/src/point.cpp File Reference . . . . .             | 148 |
| 7.61.1 Detailed Description . . . . .   | 149 |
| 7.62 /home/user/Desktop/mmm/autonomousSteeringAgents/src/prison.cpp File Reference . . . . .            | 149 |
| 7.62.1 Detailed Description . . . . .   | 150 |
| 7.62.2 Macro Definition Documentation . . . . .   | 151 |
| 7.62.2.1 DISTANCE . . . . .   | 151 |
| 7.62.2.2 WALL . . . . .   | 151 |

|  |     |
|--|-----|
| 7.63 /home/user/Desktop/mmm/autonomousSteeringAgents/src/pursuit.cpp File Reference . . . . .          | 151 |
| 7.63.1 Detailed Description . . . . .  | 152 |
| 7.64 /home/user/Desktop/mmm/autonomousSteeringAgents/src/pvector.cpp File Reference . . . . .          | 152 |
| 7.64.1 Detailed Description . . . . .  | 153 |
| 7.65 /home/user/Desktop/mmm/autonomousSteeringAgents/src/random.cpp File Reference . . . . .           | 153 |
| 7.65.1 Detailed Description . . . . .  | 154 |
| 7.66 /home/user/Desktop/mmm/autonomousSteeringAgents/src/scenario.cpp File Reference . . . . .         | 154 |
| 7.66.1 Detailed Description . . . . .  | 155 |
| 7.66.2 Macro Definition Documentation . . . . .  | 155 |
| 7.66.2.1 MAX_NUMBER_OF_AGENTS . . . . .  | 155 |
| 7.67 /home/user/Desktop/mmm/autonomousSteeringAgents/src/steeringBehavior.cpp File Reference . . . . . | 156 |
| 7.67.1 Detailed Description . . . . .  | 156 |
| 7.68 /home/user/Desktop/mmm/autonomousSteeringAgents/src/wander.cpp File Reference . . . . .           | 157 |
| 7.68.1 Detailed Description . . . . .  | 157 |
| 7.69 /home/user/Desktop/mmm/autonomousSteeringAgents/src/windy.cpp File Reference . . . . .            | 158 |
| 7.69.1 Detailed Description . . . . .  | 158 |
| 7.70 /home/user/Desktop/mmm/autonomousSteeringAgents/steeringBehavior.d File Reference . . . . .       | 159 |
| 7.71 /home/user/Desktop/mmm/autonomousSteeringAgents/test/test_suites.cpp File Reference . . . . .     | 159 |
| 7.71.1 Detailed Description . . . . .  | 160 |
| 7.71.2 Macro Definition Documentation . . . . .  | 160 |
| 7.71.2.1 BOOST_TEST_MODULE . . . . .   | 160 |
| 7.71.3 Function Documentation . . . . .  | 160 |
| 7.71.3.1 BOOST_AUTO_TEST_CASE() [1/12] . . . . .   | 161 |
| 7.71.3.2 BOOST_AUTO_TEST_CASE() [2/12] . . . . .   | 161 |
| 7.71.3.3 BOOST_AUTO_TEST_CASE() [3/12] . . . . .   | 161 |
| 7.71.3.4 BOOST_AUTO_TEST_CASE() [4/12] . . . . .   | 161 |
| 7.71.3.5 BOOST_AUTO_TEST_CASE() [5/12] . . . . .   | 162 |
| 7.71.3.6 BOOST_AUTO_TEST_CASE() [6/12] . . . . .   | 162 |
| 7.71.3.7 BOOST_AUTO_TEST_CASE() [7/12] . . . . .   | 162 |
| 7.71.3.8 BOOST_AUTO_TEST_CASE() [8/12] . . . . .   | 162 |
| 7.71.3.9 BOOST_AUTO_TEST_CASE() [9/12] . . . . .   | 163 |
| 7.71.3.10 BOOST_AUTO_TEST_CASE() [10/12] . . . . .   | 163 |
| 7.71.3.11 BOOST_AUTO_TEST_CASE() [11/12] . . . . .   | 163 |
| 7.71.3.12 BOOST_AUTO_TEST_CASE() [12/12] . . . . .   | 164 |
| 7.72 /home/user/Desktop/mmm/autonomousSteeringAgents/wander.d File Reference . . . . .                 | 164 |
| 7.73 /home/user/Desktop/mmm/autonomousSteeringAgents/windy.d File Reference . . . . .                  | 164 |



# Chapter 1

## Intent

- 1- implementing ai using autonomous steering agents
- 2- implementing smart agents using genetics algorithms
- 3- implementing smart agents using neural network

### 1.1 Dependencies

```
$sudo apt-get install libglu1-mesa-dev freeglut3-dev mesa-common-dev
```

```
$sudo apt-get install libboost-all-dev
```

### 1.2 Resources

Jan Schiffmann : Nature of Code

Fernando Bevilacqua : Understanding Steering Behaviors

Jer Thorp : Living in Data

### 1.3 Links

<https://videotutorialsrock.com/index.php>

<https://www.opengl.org/resources/libraries/glut/spec3/node1.html>

<https://learnopengl.com/Getting-started/Coordinate-Systems>





## Chapter 2

# Todo List

Member `wander::wander ()`

business logic will be changed



## Chapter 3

# Hierarchical Index

### 3.1 Class Hierarchy

This inheritance list is sorted roughly, but not completely, alphabetically:

|                             |    |
|-----------------------------|----|
| color . . . . .             | 21 |
| entity . . . . .            | 24 |
| agent . . . . .             | 13 |
| obstacle . . . . .          | 46 |
| path . . . . .              | 51 |
| flowField . . . . .         | 31 |
| graphics . . . . .          | 33 |
| point . . . . .             | 57 |
| pvector . . . . .           | 65 |
| random . . . . .            | 74 |
| scenario . . . . .          | 75 |
| evade . . . . .             | 27 |
| flee . . . . .              | 28 |
| flock . . . . .             | 30 |
| leaderFollower . . . . .    | 42 |
| mouseFollower . . . . .     | 45 |
| obstacleAvoidance . . . . . | 49 |
| pathFollower . . . . .      | 54 |
| prison . . . . .            | 62 |
| pursuit . . . . .           | 64 |
| wander . . . . .            | 89 |
| windy . . . . .             | 90 |
| steeringBehavior . . . . .  | 80 |



## Chapter 4

# Class Index

### 4.1 Class List

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

|                   |    |
|-------------------|----|
| agent             | 13 |
| color             | 21 |
| entity            | 24 |
| evade             | 27 |
| flee              | 28 |
| flock             | 30 |
| flowField         | 31 |
| graphics          | 33 |
| leaderFollower    | 42 |
| mouseFollower     | 45 |
| obstacle          | 46 |
| obstacleAvoidance | 49 |
| path              | 51 |
| pathFollower      | 54 |
| point             | 57 |
| prison            | 62 |
| pursuit           | 64 |
| pvector           | 65 |
| random            | 74 |
| scenario          | 75 |
| steeringBehavior  | 80 |
| wander            | 89 |
| windy             | 90 |



## Chapter 5

# File Index

### 5.1 File List

Here is a list of all files with brief descriptions:

|   |     |
|---|-----|
| /home/user/Desktop/mmm/autonomousSteeringAgents/agent.d . . . . .             | 93  |
| /home/user/Desktop/mmm/autonomousSteeringAgents/color.d . . . . .             | 93  |
| /home/user/Desktop/mmm/autonomousSteeringAgents/entity.d . . . . .            | 93  |
| /home/user/Desktop/mmm/autonomousSteeringAgents/evade.d . . . . .             | 93  |
| /home/user/Desktop/mmm/autonomousSteeringAgents/flee.d . . . . .              | 93  |
| /home/user/Desktop/mmm/autonomousSteeringAgents/flock.d . . . . .             | 93  |
| /home/user/Desktop/mmm/autonomousSteeringAgents/flowField.d . . . . .         | 93  |
| /home/user/Desktop/mmm/autonomousSteeringAgents/graphics.d . . . . .          | 93  |
| /home/user/Desktop/mmm/autonomousSteeringAgents/leaderFollower.d . . . . .    | 129 |
| /home/user/Desktop/mmm/autonomousSteeringAgents/main.cpp . . . . .            |     |
| Client code . . . . .   | 129 |
| /home/user/Desktop/mmm/autonomousSteeringAgents/main.d . . . . .              | 134 |
| /home/user/Desktop/mmm/autonomousSteeringAgents/mouseFollower.d . . . . .     | 134 |
| /home/user/Desktop/mmm/autonomousSteeringAgents/obstacle.d . . . . .          | 134 |
| /home/user/Desktop/mmm/autonomousSteeringAgents/obstacleAvoidance.d . . . . . | 134 |
| /home/user/Desktop/mmm/autonomousSteeringAgents/path.d . . . . .              | 134 |
| /home/user/Desktop/mmm/autonomousSteeringAgents/pathFollower.d . . . . .      | 134 |
| /home/user/Desktop/mmm/autonomousSteeringAgents/point.d . . . . .             | 134 |
| /home/user/Desktop/mmm/autonomousSteeringAgents/prison.d . . . . .            | 134 |
| /home/user/Desktop/mmm/autonomousSteeringAgents/pursuit.d . . . . .           | 134 |
| /home/user/Desktop/mmm/autonomousSteeringAgents/pvector.d . . . . .           | 134 |
| /home/user/Desktop/mmm/autonomousSteeringAgents/random.d . . . . .            | 134 |
| /home/user/Desktop/mmm/autonomousSteeringAgents/scenario.d . . . . .          | 134 |
| /home/user/Desktop/mmm/autonomousSteeringAgents/steeringBehavior.d . . . . .  | 159 |
| /home/user/Desktop/mmm/autonomousSteeringAgents/wander.d . . . . .            | 164 |
| /home/user/Desktop/mmm/autonomousSteeringAgents/windy.d . . . . .             | 164 |
| /home/user/Desktop/mmm/autonomousSteeringAgents/include/agent.h . . . . .     |     |
| Agent class defines all agent specifications . . . . .                        | 93  |
| /home/user/Desktop/mmm/autonomousSteeringAgents/include/color.h . . . . .     |     |
| Color class used for agent, path, wall etc. color . . . . .                   | 95  |
| /home/user/Desktop/mmm/autonomousSteeringAgents/include/entity.h . . . . .    | 97  |
| /home/user/Desktop/mmm/autonomousSteeringAgents/include/evade.h . . . . .     |     |
| Evade class inherited from scenario class . . . . .                           | 98  |
| /home/user/Desktop/mmm/autonomousSteeringAgents/include/flee.h . . . . .      |     |
| Agents flee from mouse scenario . . . . .                                     | 99  |

|   |     |
|---|-----|
| /home/user/Desktop/mmm/autonomousSteeringAgents/include/flock.h             |     |
| Flocking agents scenario . . . . .  | 101 |
| /home/user/Desktop/mmm/autonomousSteeringAgents/include/flowField.h         |     |
| FlowField class, screen can be filled with a force for each pixel . . . . . | 102 |
| /home/user/Desktop/mmm/autonomousSteeringAgents/include/graphics.h          |     |
| Graphics class, drives openGL . . . . .                                     | 105 |
| /home/user/Desktop/mmm/autonomousSteeringAgents/include/leaderFollower.h    |     |
| Agents follow leader scenario . . . . .                                     | 107 |
| /home/user/Desktop/mmm/autonomousSteeringAgents/include/mouseFollower.h     |     |
| Agents follow mouse scenario . . . . .                                      | 108 |
| /home/user/Desktop/mmm/autonomousSteeringAgents/include/obstacle.h          |     |
| Circular obstacles for agent avoidance behaviors . . . . .                  | 109 |
| /home/user/Desktop/mmm/autonomousSteeringAgents/include/obstacleAvoidance.h |     |
| Agents avoid from obstacles scenario . . . . .                              | 111 |
| /home/user/Desktop/mmm/autonomousSteeringAgents/include/path.h              |     |
| Path class used for path following steering behaviors . . . . .             | 112 |
| /home/user/Desktop/mmm/autonomousSteeringAgents/include/pathFollower.h      |     |
| Path following scenario . . . . .   | 113 |
| /home/user/Desktop/mmm/autonomousSteeringAgents/include/point.h             |     |
| Point class used for point operations . . . . .                             | 115 |
| /home/user/Desktop/mmm/autonomousSteeringAgents/include/prison.h            |     |
| Agents cant escape from field scenario . . . . .                            | 116 |
| /home/user/Desktop/mmm/autonomousSteeringAgents/include/pursuit.h           |     |
| One agent pursue other one scenario . . . . .                               | 117 |
| /home/user/Desktop/mmm/autonomousSteeringAgents/include/pvector.h           |     |
| Pvector class used for 2D vector operations . . . . .                       | 119 |
| /home/user/Desktop/mmm/autonomousSteeringAgents/include/random.h            |     |
| Utility class for random operations . . . . .                               | 120 |
| /home/user/Desktop/mmm/autonomousSteeringAgents/include/scenario.h          |     |
| Base class for all scenarios . . . . .                                      | 121 |
| /home/user/Desktop/mmm/autonomousSteeringAgents/include/steeringBehavior.h  |     |
| Functions for autonomous steering behaviors . . . . .                       | 122 |
| /home/user/Desktop/mmm/autonomousSteeringAgents/include/wander.h            |     |
| Random wandering agents scenario . . . . .                                  | 126 |
| /home/user/Desktop/mmm/autonomousSteeringAgents/include/windy.h             |     |
| Windy air scenario . . . . .  | 128 |
| /home/user/Desktop/mmm/autonomousSteeringAgents/src/agent.cpp               |     |
| Implementation of the agent class . . . . .                                 | 134 |
| /home/user/Desktop/mmm/autonomousSteeringAgents/src/color.cpp               |     |
| Color class implementation . . . . .  | 135 |
| /home/user/Desktop/mmm/autonomousSteeringAgents/src/entity.cpp              |     |
| Entity class implementation . . . . .                                       | 136 |
| /home/user/Desktop/mmm/autonomousSteeringAgents/src/evade.cpp               |     |
| Evade class implementation . . . . .  | 137 |
| /home/user/Desktop/mmm/autonomousSteeringAgents/src/flee.cpp                |     |
| Flee class implementation . . . . .   | 138 |
| /home/user/Desktop/mmm/autonomousSteeringAgents/src/flock.cpp               |     |
| Flock class implementation . . . . .  | 139 |
| /home/user/Desktop/mmm/autonomousSteeringAgents/src/flowField.cpp           |     |
| FlowField class implementation . . . . .                                    | 140 |
| /home/user/Desktop/mmm/autonomousSteeringAgents/src/graphics.cpp            |     |
| Graphics class implementation . . . . .                                     | 141 |
| /home/user/Desktop/mmm/autonomousSteeringAgents/src/leaderFollower.cpp      |     |
| LeaderFollower class implementation . . . . .                               | 142 |
| /home/user/Desktop/mmm/autonomousSteeringAgents/src/mouseFollower.cpp       |     |
| MouseFollower class implementation . . . . .                                | 143 |
| /home/user/Desktop/mmm/autonomousSteeringAgents/src/obstacle.cpp            |     |
| Obstacle class implementation . . . . .                                     | 144 |



|  |     |
|--|-----|
| /home/user/Desktop/mmm/autonomousSteeringAgents/src/ <a href="#">obstacleAvoidance.cpp</a> |     |
| ObstacleAvoidance class implementation . . . . .   | 145 |
| /home/user/Desktop/mmm/autonomousSteeringAgents/src/ <a href="#">path.cpp</a>              |     |
| Path class implementation . . . . .  | 146 |
| /home/user/Desktop/mmm/autonomousSteeringAgents/src/ <a href="#">pathFollower.cpp</a>      |     |
| PathFollower class implementation . . . . .  | 147 |
| /home/user/Desktop/mmm/autonomousSteeringAgents/src/ <a href="#">point.cpp</a>             |     |
| Point class implementation file . . . . .  | 148 |
| /home/user/Desktop/mmm/autonomousSteeringAgents/src/ <a href="#">prison.cpp</a>            |     |
| Prison class implementation . . . . .  | 149 |
| /home/user/Desktop/mmm/autonomousSteeringAgents/src/ <a href="#">pursuit.cpp</a>           |     |
| Prison class implementation . . . . .  | 151 |
| /home/user/Desktop/mmm/autonomousSteeringAgents/src/ <a href="#">pvector.cpp</a>           |     |
| Pvector class implementation . . . . .   | 152 |
| /home/user/Desktop/mmm/autonomousSteeringAgents/src/ <a href="#">random.cpp</a>            |     |
| Utility class for random operations . . . . .  | 153 |
| /home/user/Desktop/mmm/autonomousSteeringAgents/src/ <a href="#">scenario.cpp</a>          |     |
| Scenario base class implementation . . . . .   | 154 |
| /home/user/Desktop/mmm/autonomousSteeringAgents/src/ <a href="#">steeringBehavior.cpp</a>  |     |
| Implementation of autonomous steering behaviors . . . . .                                  | 156 |
| /home/user/Desktop/mmm/autonomousSteeringAgents/src/ <a href="#">wander.cpp</a>            |     |
| Wander class implementation . . . . .  | 157 |
| /home/user/Desktop/mmm/autonomousSteeringAgents/src/ <a href="#">windy.cpp</a>             |     |
| Windy class implementation . . . . .   | 158 |
| /home/user/Desktop/mmm/autonomousSteeringAgents/test/ <a href="#">test_suites.cpp</a>      |     |
| Unit test suites . . . . .   | 159 |



## Chapter 6

# Class Documentation

### 6.1 agent Class Reference

```
#include <agent.h>
```

#### Public Member Functions

- `agent ()`  
*default constructor.*
- `agent (float x, float y)`  
*constructor.*
- `~agent ()`  
*destructor*
- `void updatePosition (bool arrive)`  
*position update calculations*
- `void setFeatures (float s, float f, float r, float m)`  
*initialize the agent attributes*
- `string getName ()`  
*name attribute getter*
- `void setName (string n)`  
*name attribute setter*
- `float getMass ()`  
*mass attribute getter*
- `void setMass (float m)`  
*mass attribute setter*
- `void draw (graphics view) override`  
*agent drawing*
- `pvector getVelocity ()`  
*velocity attribute getter*
- `void setVelocity (pvector v)`  
*velocity attribute setter*
- `point getTarget ()`  
*target attribute getter*
- `void setTarget (point t)`  
*target attribute setter*

## Public Attributes

- float [maxSpeed](#)  
*maximum speed of the agent*
- [point position](#)  
*position of the agent*
- float [maxForce](#)  
*maximum force of the agent*
- [pvector steering](#)  
*steering force of the apply*
- [pvector force](#)  
*force to apply to the agent*
- [pvector acceleration](#)  
*acceleration of the agent*
- [pvector desiredVelocity](#)  
*desired velocity of the agent*
- float [r](#)  
*radius for arrive behavior*
- int [id](#)  
*id of the agent*
- bool [arrive](#) = false  
*has arriving behavior or not*

### 6.1.1 Detailed Description

Definition at line 21 of file agent.h.

### 6.1.2 Constructor & Destructor Documentation

#### 6.1.2.1 [agent\(\)](#) [1/2]

```
agent::agent ( )
```

default constructor.

See also

[agent\(float x, float y\)](#)

Definition at line 16 of file agent.cpp.

```
17 {  
18  
19 }
```

#### 6.1.2.2 [agent\(\)](#) [2/2]

```
agent::agent (  
    float x,  
    float y )
```

constructor.

## Parameters

|          |                         |
|----------|-------------------------|
| <i>x</i> | position x of the agent |
| <i>y</i> | position y of the agent |

## See also

[agent\(\)](#)

Definition at line 37 of file agent.cpp.

```
38 {  
39     position      = point(x, y);  
40     velocity      = pvector(0.6, 0.0);  
41     acceleration  = pvector(0.0, 0.0);  
42     steering      = pvector(0.0, 0.0);  
43     desiredVelocity = pvector(0.0, 0.0);  
44     force         = pvector(0.0, 0.0);  
45     targetPoint   = point(0.0, 0.0);  
46     setColor(RED);  
47 }
```

### 6.1.2.3 ~agent()

```
agent::~agent ( )
```

destructor

Definition at line 96 of file agent.cpp.

```
97 {  
98  
99 }
```

## 6.1.3 Member Function Documentation

### 6.1.3.1 draw()

```
void agent::draw (  
    graphics view ) [override], [virtual]
```

agent drawing

## Parameters

|             |                  |
|-------------|------------------|
| <i>view</i> | graphics to draw |
|-------------|------------------|

Implements [entity](#).

Definition at line 101 of file agent.cpp.

```
101         {
102     this->updatePosition(this->arrive);
103     view.drawAgent(*this);
104 }
```

#### 6.1.3.2 getMass()

```
float agent::getMass ( )
```

mass attribute getter

Definition at line 29 of file agent.cpp.

```
29     {
30     return mass;
31 }
```

#### 6.1.3.3 getName()

```
string agent::getName ( )
```

name attribute getter

Definition at line 21 of file agent.cpp.

```
21     {
22     return name;
23 }
```

#### 6.1.3.4 getTarget()

```
point agent::getTarget ( )
```

target attribute getter

Definition at line 57 of file agent.cpp.

```
57     {
58     return targetPoint;
59 }
```

#### 6.1.3.5 getVelocity()

```
pvector agent::getVelocity ( )
```

velocity attribute getter

Definition at line 49 of file agent.cpp.

```
49     {
50     return velocity;
51 }
```

#### 6.1.3.6 setFeatures()

```
void agent::setFeatures (
    float s,
    float f,
    float r,
    float m )
```

initialize the agent attributes

**Parameters**

|          |                              |
|----------|------------------------------|
| <i>s</i> | maximum velocity             |
| <i>f</i> | maximum force                |
| <i>r</i> | radius for arriving behavior |
| <i>m</i> | mass                         |

Definition at line 88 of file agent.cpp.

```
89 {  
90     this->maxSpeed = s;  
91     this->maxForce = f;  
92     this->r = r;  
93     this->mass = m;  
94 }
```

**6.1.3.7 setMass()**

```
void agent::setMass (  
    float m )
```

mass attribute setter

**Parameters**

|          |           |
|----------|-----------|
| <i>m</i> | set value |
|----------|-----------|

Definition at line 33 of file agent.cpp.

```
33     {  
34         mass = m;  
35     }
```

**6.1.3.8 setName()**

```
void agent::setName (  
    string n )
```

name attribute setter

**Parameters**

|          |           |
|----------|-----------|
| <i>n</i> | set value |
|----------|-----------|

Definition at line 25 of file agent.cpp.

```
25     {  
26         name = n;  
27     }
```

### 6.1.3.9 setTarget()

```
void agent::setTarget (
    point t )
```

target attribute setter

#### Parameters

|          |           |
|----------|-----------|
| <i>t</i> | set value |
|----------|-----------|

Definition at line 61 of file agent.cpp.

```
61     {
62         targetPoint = t;
63     }
```

### 6.1.3.10 setVelocity()

```
void agent::setVelocity (
    pvector v )
```

velocity attribute setter

#### Parameters

|          |           |
|----------|-----------|
| <i>v</i> | set value |
|----------|-----------|

Definition at line 53 of file agent.cpp.

```
53     {
54         velocity = v;
55     }
```

### 6.1.3.11 updatePosition()

```
void agent::updatePosition (
    bool arrive )
```

position update calculations

#### Parameters

|               |                              |
|---------------|------------------------------|
| <i>arrive</i> | has arriving behavior or not |
|---------------|------------------------------|

Definition at line 66 of file agent.cpp.

```
67 {
68     force.limit(maxForce);
69     acceleration = force;
70     velocity += acceleration;
71 }
```



```
72     //arriving behavior implementation
73     if(arrive == true){
74         pvector diff = targetPoint - position;
75         if(diff.magnitude() > r)
76             velocity.limit(maxSpeed);
77         else
78             velocity.limit(maxSpeed * diff.magnitude() / r);
79     }
80     else{
81         velocity.limit(maxSpeed);
82     }
83
84     position = position + velocity;
85     force = pvector(0,0);
86 }
```

## 6.1.4 Member Data Documentation

### 6.1.4.1 acceleration

`pvector agent::acceleration`

acceleration of the agent

Definition at line 135 of file agent.h.

### 6.1.4.2 arrive

`bool agent::arrive = false`

has arriving behavior or not

Definition at line 155 of file agent.h.

### 6.1.4.3 desiredVelocity

`pvector agent::desiredVelocity`

desired velocity of the agent

Definition at line 140 of file agent.h.

#### 6.1.4.4 force

```
pvector agent::force
```

force to apply to the agent

Definition at line 130 of file agent.h.

#### 6.1.4.5 id

```
int agent::id
```

id of the agent

Definition at line 150 of file agent.h.

#### 6.1.4.6 maxForce

```
float agent::maxForce
```

maximum force of the agent

Definition at line 120 of file agent.h.

#### 6.1.4.7 maxSpeed

```
float agent::maxSpeed
```

maximum speed of the agent

Definition at line 110 of file agent.h.

#### 6.1.4.8 position

```
point agent::position
```

position of the agent

Definition at line 115 of file agent.h.

#### 6.1.4.9 r

```
float agent::r
```

radius for arrive behavior

Definition at line 145 of file agent.h.

#### 6.1.4.10 steering

```
pvector agent::steering
```

steering force of the apply

Definition at line 125 of file agent.h.

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

- [/home/user/Desktop/mmm/autonomousSteeringAgents/include/agent.h](#)
- [/home/user/Desktop/mmm/autonomousSteeringAgents/src/agent.cpp](#)

## 6.2 color Class Reference

```
#include <color.h>
```

### Public Member Functions

- [color](#) ()  
*default constructor.*
- [color](#) (float r, float g, float b)  
*constructor.*

### Static Public Member Functions

- static [color](#) [getColor](#) (int index)  
*gets colorbar colors*

### Public Attributes

- float [R](#)  
*portion of red color*
- float [G](#)  
*portion of green color*
- float [B](#)  
*portion of blue color*

## 6.2.1 Detailed Description

Definition at line 23 of file color.h.

## 6.2.2 Constructor & Destructor Documentation

### 6.2.2.1 color() [1/2]

```
color::color ( )
```

default constructor.

See also

[color\(float r, float g, float b\)](#)

Definition at line 13 of file color.cpp.

```
14 {  
15  
16 }
```

### 6.2.2.2 color() [2/2]

```
color::color (  
    float r,  
    float g,  
    float b )
```

constructor.

Parameters

|          |               |
|----------|---------------|
| <i>r</i> | red (0-255)   |
| <i>g</i> | green (0-255) |
| <i>b</i> | blue (0-255)  |

See also

[path\(\)](#)

Definition at line 18 of file color.cpp.

```
19 {  
20     R = r;  
21     G = g;  
22     B = b;  
23 }
```

## 6.2.3 Member Function Documentation

### 6.2.3.1 getColor()

```
color color::getColor (
    int index ) [static]
```

gets colorbar colors

#### Parameters

|              |          |
|--------------|----------|
| <i>index</i> | color id |
|--------------|----------|

Definition at line 25 of file color.cpp.

```
25 {
26     switch (index)
27     {
28         case 0: return WHITE; break;
29         case 1: return BLUE; break;
30         case 2: return RED; break;
31         case 3: return YELLOW; break;
32         case 4: return GREEN; break;
33         case 5: return BLACK; break;
34         case 6: return CYAN; break;
35         case 7: return MAGENDA; break;
36     }
37     return RED;
38 }
39 }
```

## 6.2.4 Member Data Documentation

### 6.2.4.1 B

```
float color::B
```

portion of blue color

Definition at line 53 of file color.h.

### 6.2.4.2 G

```
float color::G
```

portion of green color

Definition at line 48 of file color.h.

### 6.2.4.3 R

```
float color::R
```

portion of red color

Definition at line 43 of file color.h.

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

- [/home/user/Desktop/mmm/autonomousSteeringAgents/include/color.h](#)
- [/home/user/Desktop/mmm/autonomousSteeringAgents/src/color.cpp](#)

## 6.3 entity Class Reference

```
#include <entity.h>
```

### Public Member Functions

- [entity](#) ()  
*default constructor.*
- string [getName](#) ()  
*getter of the name*
- void [setName](#) (string name)  
*name attribute setter*
- int [getId](#) ()  
*getter of the id attribute*
- void [setId](#) (int id)  
*id attribute setter*
- virtual void [draw](#) ([graphics](#) view)=0  
*overriden by child classes*
- [color](#) [getColor](#) ()  
*getter of the color attribute*
- void [setColor](#) ([color](#) color)  
*getter of the color attribute*

### 6.3.1 Detailed Description

Definition at line 10 of file entity.h.

### 6.3.2 Constructor & Destructor Documentation

### 6.3.2.1 entity()

```
entity::entity ( )
```

default constructor.

Definition at line 10 of file entity.cpp.

```
10     {  
11         entityColor = RED;  
12     }
```

## 6.3.3 Member Function Documentation

### 6.3.3.1 draw()

```
virtual void entity::draw (  
    graphics view ) [pure virtual]
```

overridden by child classes

#### Parameters

|             |          |
|-------------|----------|
| <i>view</i> | graphics |
|-------------|----------|

Implemented in [agent](#), [obstacle](#), and [path](#).

### 6.3.3.2 getColor()

```
color entity::getColor ( )
```

getter of the color attribute

Definition at line 15 of file entity.cpp.

```
16 {  
17     return entityColor;  
18 }
```

### 6.3.3.3 getId()

```
int entity::getId ( )
```

getter of the id attribute

Definition at line 33 of file entity.cpp.

```
33     {  
34         return id;  
35     }
```

#### 6.3.3.4 getName()

```
string entity::getName ( )
```

getter of the name

Definition at line 25 of file entity.cpp.

```
25     {  
26         return name;  
27     }
```

#### 6.3.3.5 setColor()

```
void entity::setColor (  
    color color )
```

getter of the color attribute

Definition at line 20 of file entity.cpp.

```
20     {  
21         entityColor = color;  
22     }
```

#### 6.3.3.6 setId()

```
void entity::setId (  
    int id )
```

id attribute setter

##### Parameters

|           |        |
|-----------|--------|
| <i>id</i> | setter |
|-----------|--------|

Definition at line 37 of file entity.cpp.

```
37     {  
38         this->id = id;  
39     }
```

#### 6.3.3.7 setName()

```
void entity::setName (  
    string name )
```

name attribute setter



## Parameters

|             |        |
|-------------|--------|
| <i>name</i> | setter |
|-------------|--------|

Definition at line 29 of file entity.cpp.

```

29      {
30          this->name = name;
31      }

```

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

- /home/user/Desktop/mmm/autonomousSteeringAgents/include/[entity.h](#)
- /home/user/Desktop/mmm/autonomousSteeringAgents/src/[entity.cpp](#)

## 6.4 evade Class Reference

```
#include <evade.h>
```

### Public Member Functions

- [evade](#) ()  
*default constructor.*

### Static Public Member Functions

- static void [loop](#) ()  
*loop function of evading scenario*

### Additional Inherited Members

#### 6.4.1 Detailed Description

Definition at line 15 of file evade.h.

#### 6.4.2 Constructor & Destructor Documentation

##### 6.4.2.1 evade()

```
evade::evade ( )
```

default constructor.

Definition at line 31 of file evade.cpp.

```

32 {
33     name = "evading";
34     createStaticAgents("gazelle", "lion");
35     callback = reinterpret_cast<void(*)()> ( (void *)(&loop) );
36 }

```

## 6.4.3 Member Function Documentation

### 6.4.3.1 loop()

```
void evade::loop ( ) [static]
```

loop function of evading scenario

#### Note

opengl callback forces that function to be static

Definition at line 15 of file evade.cpp.

```
16 {  
17     for(auto it = agents.begin(); it < agents.end(); it++){  
18         if((*it).getName() == "lion"){  
19             (*it).setTarget(view.getMousePosition());  
20             (*it).force = behavior.seek(*it);  
21             (*it).arrive = true;  
22         }  
23         else{//gazelle  
24             (*it).force = behavior.evade(agents, *it, view, "lion");  
25         }  
26     }  
27  
28     refresh();  
29 }
```

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

- /home/user/Desktop/mmm/autonomousSteeringAgents/include/[evade.h](#)
- /home/user/Desktop/mmm/autonomousSteeringAgents/src/[evade.cpp](#)

## 6.5 flee Class Reference

```
#include <flee.h>
```

### Public Member Functions

- [flee](#) ()  
*default constructor.*

### Static Public Member Functions

- static void [loop](#) ()  
*evading scenario loop function*

## Additional Inherited Members

### 6.5.1 Detailed Description

Definition at line 14 of file flee.h.

### 6.5.2 Constructor & Destructor Documentation

#### 6.5.2.1 flee()

```
flee::flee ( )
```

default constructor.

Definition at line 24 of file flee.cpp.

```
25 {
26     int agentCount = 196;
27     name = "fleeing troop";
28     createTroop(agentCount);
29     callback = reinterpret_cast <void(*)()> ( (void *)(&loop) );
30 }
```

### 6.5.3 Member Function Documentation

#### 6.5.3.1 loop()

```
void flee::loop ( ) [static]
```

evading scenario loop function

#### Note

opengl callback forces that function to be static

Definition at line 15 of file flee.cpp.

```
16 {
17     for(auto it = agents.begin(); it < agents.end(); it++){
18         (*it).force = behavior.flee((*it), view, view.getMousePosition());
19     }
20
21     refresh();
22 }
```

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

- /home/user/Desktop/mmm/autonomousSteeringAgents/include/flee.h
- /home/user/Desktop/mmm/autonomousSteeringAgents/src/flee.cpp

## 6.6 flock Class Reference

```
#include <flock.h>
```

### Public Member Functions

- `flock ()`  
*default constructor.*

### Static Public Member Functions

- static void `loop ()`  
*flocking scenario loop function*

### Additional Inherited Members

#### 6.6.1 Detailed Description

Definition at line 15 of file flock.h.

#### 6.6.2 Constructor & Destructor Documentation

##### 6.6.2.1 flock()

```
flock::flock ( )
```

default constructor.

Definition at line 32 of file flock.cpp.

```
33 {  
34     int agentCount = 40;  
35     float maxForce = 0.5;  
36     float maxSpeed = 0.9;  
37     name = "flocking agents";  
38  
39     createRandomAgents(agentCount, maxForce, maxSpeed);  
40     callback = reinterpret_cast<void(*)()> ( (void *)(&loop) );  
41 }
```

#### 6.6.3 Member Function Documentation

### 6.6.3.1 loop()

```
void flock::loop ( ) [static]
```

flocking scenario loop function

#### Note

opengl callback forces that function to be static

Definition at line 15 of file flock.cpp.

```
16 {
17     for(auto it = agents.begin(); it < agents.end(); it++){
18         view.forceInScreen((*it));
19     }
20     pvector sep = behavior.separation(agents, *it, 6);
21     sep.mul(1);
22     pvector ali = behavior.align(agents, *it, 20);
23     ali.mul(4);
24     pvector coh = behavior.cohesion(agents, *it, 20);
25     coh.mul(0.1);
26
27     (*it).force = sep + ali + coh;
28 }
29 refresh();
30 }
```

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

- [/home/user/Desktop/mmm/autonomousSteeringAgents/include/flock.h](#)
- [/home/user/Desktop/mmm/autonomousSteeringAgents/src/flock.cpp](#)

## 6.7 flowField Class Reference

```
#include <flowField.h>
```

### Public Member Functions

- [flowField \(\)](#)  
*default constructor.*
- [flowField \(pvector p\)](#)  
*constructor.*
- [pvector getField \(int x, int y\)](#)  
*get force at individual pixel*

### 6.7.1 Detailed Description

Definition at line 18 of file flowField.h.

### 6.7.2 Constructor & Destructor Documentation

### 6.7.2.1 flowField() [1/2]

```
flowField::flowField ( )
```

default constructor.

See also

[flowField\(pvector p\)](#)

Definition at line 15 of file flowField.cpp.

```
16 {  
17  
18 }
```

### 6.7.2.2 flowField() [2/2]

```
flowField::flowField (  
    pvector p )
```

constructor.

Parameters

|          |              |
|----------|--------------|
| <i>p</i> | force vector |
|----------|--------------|

See also

[flowField\(\)](#)

Definition at line 10 of file flowField.cpp.

```
11 {  
12     createFlowField(p);  
13 }
```

## 6.7.3 Member Function Documentation

### 6.7.3.1 getField()

```
pvector flowField::getField (  
    int x,  
    int y )
```

get force at individual pixel

## Parameters

|          |            |
|----------|------------|
| <i>x</i> | coordinate |
| <i>y</i> | coordinate |

## Returns

force at specified position

Definition at line 39 of file flowField.cpp.

```
40 {
41     return uniformField[x][y];
42 }
```

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

- /home/user/Desktop/mmm/autonomousSteeringAgents/include/flowField.h
- /home/user/Desktop/mmm/autonomousSteeringAgents/src/flowField.cpp

## 6.8 graphics Class Reference

```
#include <graphics.h>
```

### Public Member Functions

- void [drawAgent](#) ([agent](#) &[agent](#))  
*drawing with corresponding angle*
- void [drawLine](#) ([point](#) p1, [point](#) p2, [color](#) cl)  
*drawing line*
- void [drawPath](#) ([path](#) &[path](#))  
*draws path*
- void [drawPoint](#) ([point](#) p)  
*draws point*
- void [drawCircle](#) ([point](#) p, float radius, [color](#) color)  
*draws circle*
- void [drawText](#) (string text, [point](#) p)  
*draws text on screen*
- void [forceInScreen](#) ([agent](#) &[agent](#))  
*changes agent position so that it stays in screen*
- void [refreshScene](#) ()  
*update agent position*
- [point](#) [getMousePosition](#) ()  
*gets mouse position*
- void [initGraphics](#) (int \*argv, char \*\*argc, void(\*callback)())  
*initialization of graphics*

## Static Public Member Functions

- static void `timerEvent` (int value)  
*periodic timer event*
- static void `handleKeyPress` (unsigned char key, int x, int y)  
*key press event*
- static void `mouseButton` (int button, int state, int x, int y)  
*mouse press event*
- static void `handleResize` (int w, int h)  
*event triggered with screen resizing*
- static void `mouseMove` (int x, int y)  
*event triggered with mouse movements*

## Static Public Attributes

- static int `target_x` = `-WIDTH`  
*mouse position x*
- static int `target_y` = `HEIGHT`  
*mouse position y*

### 6.8.1 Detailed Description

Definition at line 22 of file `graphics.h`.

### 6.8.2 Member Function Documentation

#### 6.8.2.1 `drawAgent()`

```
void graphics::drawAgent (
    agent & agent )
```

drawing with corresponding angle

#### Parameters

|              |                    |
|--------------|--------------------|
| <i>agent</i> | instance to change |
|--------------|--------------------|

Definition at line 159 of file `graphics.cpp`.

```
160 {
161     glPushMatrix();
162     glTranslatef(agent.position.x, agent.position.y, 0.0f);
163     glRotatef(agent.getVelocity().getAngle(), 0.0f, 0.0f, 1.0f);
164     glBegin(GL_TRIANGLES);
165     glColor3f( agent.getColor().R, agent.getColor().G, agent.getColor().B);
166     glVertex3f( 1.0f, 0.0f, 0.0f);
167     glVertex3f(-1.0f, 0.5f, 0.0f);
168     glVertex3f(-1.0f, -0.5f, 0.0f);
169     glEnd();
170     glPopMatrix();
```



```
171 }
```

### 6.8.2.2 drawCircle()

```
void graphics::drawCircle (
    point p,
    float radius,
    color color )
```

draws circle

#### Parameters

|               |                      |
|---------------|----------------------|
| <i>p</i>      | center of the circle |
| <i>radius</i> | radius of the circle |
| <i>color</i>  | of the circle        |

Definition at line 136 of file graphics.cpp.

```
137 {
138     glColor3f(color.R, color.G, color.B);
139     glBegin(GL_LINE_STRIP);
140     glLineWidth(2);
141     for (int i = 0; i <= 300; i++) {
142         float angle = 2 * PI * i / 300;
143         float x = cos(angle) * radius;
144         float y = sin(angle) * radius;
145         glVertex2d(p.x + x, p.y + y);
146     }
147     glEnd();
148 }
```

### 6.8.2.3 drawLine()

```
void graphics::drawLine (
    point p1,
    point p2,
    color cl )
```

drawing line

#### Parameters

|           |                         |
|-----------|-------------------------|
| <i>p1</i> | start point of the line |
| <i>p2</i> | end point of the line   |
| <i>cl</i> | color of the line       |

Definition at line 126 of file graphics.cpp.

```
127 {
128     glColor3f( cl.R, cl.G, cl.B);
129     glLineWidth(2);
130     glBegin(GL_LINES);
131     glVertex2f(p1.x, p1.y);
132     glVertex2f(p2.x, p2.y);
```

```

133     glEnd();
134 }

```

#### 6.8.2.4 drawPath()

```

void graphics::drawPath (
    path & path )

```

draws path

##### Parameters

|             |         |
|-------------|---------|
| <i>path</i> | to draw |
|-------------|---------|

Definition at line 112 of file graphics.cpp.

```

113 {
114     point p1, p2;
115     for(auto it = path.points.begin(); it < path.points.end()-1; it++){
116         p1 = point((*it).x, (*it).y - path.getPathWidth() / 2) ;
117         p2 = point((*it+1).x, (*it+1).y - path.getPathWidth() / 2);
118         drawLine(p1, p2, path.getColor());
119
120         p1 = point((*it).x, (*it).y + path.getPathWidth() / 2) ;
121         p2 = point((*it+1).x, (*it+1).y + path.getPathWidth() / 2);
122         drawLine(p1, p2, path.getColor());
123     }
124 }

```

#### 6.8.2.5 drawPoint()

```

void graphics::drawPoint (
    point p )

```

draws point

##### Parameters

|          |               |
|----------|---------------|
| <i>p</i> | point to draw |
|----------|---------------|

Definition at line 150 of file graphics.cpp.

```

151 {
152     glColor3f(1,1,1);
153     glPointSize(4.0);
154     glBegin(GL_POINTS);
155     glVertex2f(p.x, p.y);
156     glEnd();
157 }

```

#### 6.8.2.6 drawText()

```

void graphics::drawText (
    string text,
    point p )

```

draws text on screen

#### Parameters

|             |                      |
|-------------|----------------------|
| <i>p</i>    | position of the text |
| <i>text</i> | to display           |

Definition at line 21 of file graphics.cpp.

```

22 {
23     glColor3f (0.0, 0.0, 1.0);
24     glRasterPos2f(p.x, p.y);
25     for ( string::iterator it=text.begin(); it!=text.end(); ++it){
26         glutBitmapCharacter(GLUT_BITMAP_9_BY_15, *it);
27     }
28 }
```

#### 6.8.2.7 forceInScreen()

```

void graphics::forceInScreen (
    agent & agent )
```

changes agent position so that it stays in screen

#### Parameters

|              |          |
|--------------|----------|
| <i>agent</i> | instance |
|--------------|----------|

Definition at line 61 of file graphics.cpp.

```

62 {
63     if (agent.position.x > WIDTH)
64         agent.position.x -= 2 * WIDTH;
65     if (agent.position.x < -WIDTH)
66         agent.position.x += 2 * WIDTH;
67     if (agent.position.y > HEIGHT)
68         agent.position.y -= 2 * HEIGHT;
69     if (agent.position.y < -HEIGHT)
70         agent.position.y += 2 * HEIGHT;
71 }
```

#### 6.8.2.8 getMousePosition()

```

point graphics::getMousePosition ( )
```

gets mouse position

#### Returns

mouse point

Definition at line 56 of file graphics.cpp.

```

57 {
58     return point (graphics::target_x, graphics::target_y);
59 }
```

### 6.8.2.9 handleKeypress()

```
void graphics::handleKeypress (
    unsigned char key,
    int x,
    int y ) [static]
```

key press event

#### Parameters

|            |                                |
|------------|--------------------------------|
| <i>key</i> | pressed                        |
| <i>x</i>   | unused but required for OpenGL |
| <i>y</i>   | unused but required for OpenGL |

Definition at line 105 of file graphics.cpp.

```
106 {
107     if (key == ESC){
108         exit(0);
109     }
110 }
```

### 6.8.2.10 handleResize()

```
void graphics::handleResize (
    int w,
    int h ) [static]
```

event triggered with screen resizing

#### Parameters

|          |                      |
|----------|----------------------|
| <i>w</i> | width of the screen  |
| <i>h</i> | height of the screen |

Definition at line 81 of file graphics.cpp.

```
82 {
83     glViewport(0, 0, w, h); //Tell OpenGL how to convert from coordinates to pixel values
84     glMatrixMode(GL_PROJECTION); //Switch to setting the camera perspective
85     glLoadIdentity(); //Reset the camera
86     //Set the camera perspective
87     gluPerspective(45.0, //The camera angle
88                   (double)w / (double)h, //The width-to-height ratio
89                   1.0, //The near z clipping coordinate
90                   200.0); //The far z clipping coordinate
91 }
```

### 6.8.2.11 initGraphics()

```
void graphics::initGraphics (
    int * argv,
```

```
char ** argc,  
void(*)() callback )
```

initialization of graphics

## Parameters

|                 |  |
|-----------------|--|
| <i>argv</i>     | user parameters                            |
| <i>argc</i>     | count of user parameters                   |
| <i>callback</i> | loop function for openGL periodic callback |

Definition at line 39 of file graphics.cpp.

```

40 {
41     glutInit(argv, argc);
42     glutInitDisplayMode(GLUT_DOUBLE | GLUT_RGB | GLUT_DEPTH);
43     glutInitWindowSize(400, 400);
44     glutCreateWindow("Autonomous Steering Agents");
45     glClearColor(0.7f, 0.7f, 0.7f, 1.0f); //set background color
46     glEnable(GL_DEPTH_TEST);
47     glutDisplayFunc(*callback);
48     glutMouseFunc(graphics::mouseButton);
49     glutPassiveMotionFunc(graphics::mouseMove);
50     glutKeyboardFunc(graphics::handleKeypress);
51     glutReshapeFunc(graphics::handleResize);
52     glutTimerFunc(20, graphics::timerEvent, 0);
53     glutMainLoop();
54 }
```

### 6.8.2.12 mouseButton()

```

void graphics::mouseButton (
    int button,
    int state,
    int x,
    int y ) [static]
```

mouse press event

## Parameters

|               |                                |
|---------------|--------------------------------|
| <i>button</i> | mouse key pressed              |
| <i>state</i>  | down/up etc.                   |
| <i>x</i>      | unused but required for openGL |
| <i>y</i>      | unused but required for openGL |

Definition at line 99 of file graphics.cpp.

```

100 {
101     if (button == GLUT_LEFT_BUTTON && state == GLUT_DOWN){
102     }
103 }
```

### 6.8.2.13 mouseMove()

```

void graphics::mouseMove (
    int x,
    int y ) [static]
```

event triggered with mouse movements

## Parameters

|          |                       |
|----------|-----------------------|
| <i>x</i> | osition of the mouse  |
| <i>y</i> | position of the mouse |

Definition at line 73 of file graphics.cpp.

```

74 {
75     //TODO: mouse position to glut
76     //TODO: magic numbers
77     graphics::target_x = x / 5.88 - 34;
78     graphics::target_y = 34 - y / 5.88;
79 }
```

### 6.8.2.14 refreshScene()

```
void graphics::refreshScene ( )
```

update agent position

Definition at line 30 of file graphics.cpp.

```

31 {
32     glutSwapBuffers();
33     glClear(GL_COLOR_BUFFER_BIT | GL_DEPTH_BUFFER_BIT);
34     glMatrixMode(GL_MODELVIEW); //Switch to the drawing perspective
35     glLoadIdentity(); //Reset the drawing perspective
36     glTranslatef(0.0f, 0.0f, -85.0f); //Move to the center of the triangle
37 }
```

### 6.8.2.15 timerEvent()

```
void graphics::timerEvent (
    int value ) [static]
```

periodic timer event

## Parameters

|              |              |
|--------------|--------------|
| <i>value</i> | period as ms |
|--------------|--------------|

Definition at line 93 of file graphics.cpp.

```

94 {
95     glutPostRedisplay(); //Tell GLUT that the display has changed
96     glutTimerFunc(value, timerEvent, 20);
97 }
```

## 6.8.3 Member Data Documentation

### 6.8.3.1 target\_x

```
int graphics::target_x = -WIDTH [static]
```

mouse position x

Definition at line 130 of file graphics.h.

### 6.8.3.2 target\_y

```
int graphics::target_y = HEIGHT [static]
```

mouse position y

Definition at line 135 of file graphics.h.

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

- [/home/user/Desktop/mmm/autonomousSteeringAgents/include/graphics.h](#)
- [/home/user/Desktop/mmm/autonomousSteeringAgents/src/graphics.cpp](#)

## 6.9 leaderFollower Class Reference

```
#include <leaderFollower.h>
```

### Public Member Functions

- [leaderFollower](#) ()  
*default constructor.*

### Static Public Member Functions

- static void [loop](#) ()  
*leader following scenario loop function*

### Static Public Attributes

- static [pvector](#) [leaderVelocity](#)  
*leader velocity*
- static [point](#) [leaderPosition](#)  
*leader position*



## Additional Inherited Members

### 6.9.1 Detailed Description

Definition at line 14 of file leaderFollower.h.

### 6.9.2 Constructor & Destructor Documentation

#### 6.9.2.1 leaderFollower()

```
leaderFollower::leaderFollower ( )
```

default constructor.

Definition at line 61 of file leaderFollower.cpp.

```
62 {  
63     int agentCount = 4;  
64     float maxForce = 0.4;  
65     float maxSpeed = 0.4;  
66     name = "leader following";  
67  
68     //todo: refactor leader creation  
69     agent agent1 {-10.0, 0.0};  
70     agent1.id = 1;  
71     agent1.setName("leader");  
72     agent1.setColor(BLUE);  
73     agent1.setFeatures(0.8, 0.4, 5, 1);  
74     agents.push_back(agent1);  
75  
76     createRandomAgents(agentCount, maxForce, maxSpeed);  
77     callback = reinterpret_cast <void(*)()> ( (void *)(&loop) );  
78 }
```

### 6.9.3 Member Function Documentation

#### 6.9.3.1 loop()

```
void leaderFollower::loop ( ) [static]
```

leader following scenario loop function

**Note**

opengl callback forces that function to be static

Definition at line 18 of file leaderFollower.cpp.

```

19 {
20     for(auto it = agents.begin(); it < agents.end(); it++){
21         if((*it).getName() == "leader"){
22             (*it).setTarget(view.getMousePosition());
23             (*it).force = behavior.seek(*it);
24             leaderVelocity = (*it).getVelocity();
25             leaderVelocity.mul(-1);
26             leaderVelocity.normalize().mul(20);
27             leaderPosition = (*it).position;
28
29             view.drawText((*it).getName(), point(leaderPosition.x - 3, leaderPosition.y - 3));
30
31         }
32         else{
33             pvector sep = behavior.separation(agents, *it, 3);
34             sep.mul(15);
35             (*it).force = sep;
36
37             (*it).setTarget(leaderPosition + leaderVelocity);
38             view.drawCircle((*it).getTarget(), 8, RED);
39
40             /*
41             pvector leaderDiff = (*it).position - leaderPosition;
42             if(leaderDiff.magnitude() < 5){
43                 pvector fle = behavior.evade(agents, *it, view, "leader");
44                 fle.mul(40);
45                 (*it).force += fle;
46             }*/
47
48             pvector diff = (*it).position - (*it).getTarget();
49             if(diff.magnitude() > 5) {
50                 (*it).force += behavior.seek(*it);
51             }
52             else{
53                 (*it).setVelocity(pvector(0,0));
54             }
55         }
56         (*it).arrive = true;
57     }
58     refresh();
59 }

```

## 6.9.4 Member Data Documentation

### 6.9.4.1 leaderPosition

`point` leaderFollower::leaderPosition [static]

leader position

Definition at line 24 of file leaderFollower.h.

### 6.9.4.2 leaderVelocity

`pvector` leaderFollower::leaderVelocity [static]

leader velocity

Definition at line 19 of file leaderFollower.h.

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

- `/home/user/Desktop/mmm/autonomousSteeringAgents/include/leaderFollower.h`
- `/home/user/Desktop/mmm/autonomousSteeringAgents/src/leaderFollower.cpp`

## 6.10 mouseFollower Class Reference

```
#include <mouseFollower.h>
```

### Public Member Functions

- [mouseFollower](#) ()  
*default constructor.*

### Static Public Member Functions

- static void [loop](#) ()  
*mouse following scenario loop function*

### Additional Inherited Members

#### 6.10.1 Detailed Description

Definition at line 14 of file mouseFollower.h.

#### 6.10.2 Constructor & Destructor Documentation

##### 6.10.2.1 mouseFollower()

```
mouseFollower::mouseFollower ( )
```

default constructor.

Definition at line 25 of file mouseFollower.cpp.

```
26 {  
27     int agentCount = 30;  
28     float maxForce = 0.3;  
29     float maxSpeed = 0.6;  
30     name = "mouse following";  
31     createRandomAgents(agentCount, maxForce, maxSpeed);  
32     callback = reinterpret_cast<void(*)()> ( (void *)(&loop) );  
33 }
```

#### 6.10.3 Member Function Documentation

### 6.10.3.1 loop()

```
void mouseFollower::loop ( ) [static]
```

mouse following scenario loop function

#### Note

opengl callback forces that function to be static

Definition at line 15 of file mouseFollower.cpp.

```
16 {
17     for(auto it = agents.begin(); it < agents.end(); it++){
18         (*it).setTarget(view.getMousePosition());
19         (*it).force = behavior.seek(*it);
20         (*it).arrive = true;
21     }
22     refresh();
23 }
```

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

- [/home/user/Desktop/mmm/autonomousSteeringAgents/include/mouseFollower.h](#)
- [/home/user/Desktop/mmm/autonomousSteeringAgents/src/mouseFollower.cpp](#)

## 6.11 obstacle Class Reference

```
#include <obstacle.h>
```

### Public Member Functions

- [obstacle](#) ()  
*default constructor.*
- [obstacle](#) ([point](#) p, float r)  
*constructor*
- [point](#) [getCenter](#) ()  
*getter of the center point attribute*
- void [setCenter](#) ([point](#) p)  
*setter of the center point attribute*
- float [getRadius](#) ()  
*getter of the radius attribute*
- void [setRadius](#) (float r)  
*setter of the radius attribute*
- void [draw](#) ([graphics](#) view) override  
*overriden draw implementation*

### 6.11.1 Detailed Description

Definition at line 15 of file obstacle.h.

## 6.11.2 Constructor & Destructor Documentation

### 6.11.2.1 obstacle() [1/2]

```
obstacle::obstacle ( )
```

default constructor.

See also

[obstacle\(point p, float r](#)

Definition at line 16 of file obstacle.cpp.

```
17 {
18     p = point(0,0);
19     r = 5;
20     setColor(RED);
21 }
```

### 6.11.2.2 obstacle() [2/2]

```
obstacle::obstacle (
    point p,
    float r )
```

constructor

Parameters

|     |                                 |
|-----|---------------------------------|
| $p$ | center of the circular obstacle |
| $r$ | radius of the obstacle          |

See also

[obstacle\(point p, float r\);](#)

Definition at line 23 of file obstacle.cpp.

```
24 {
25     this->p = p;
26     this->r = r;
27     setColor(RED);
28 }
```

## 6.11.3 Member Function Documentation

### 6.11.3.1 draw()

```
void obstacle::draw (
    graphics view ) [override], [virtual]
```

overriden draw implementation

Implements [entity](#).

Definition at line 30 of file obstacle.cpp.

```
30     {
31     view.drawCircle(p, r, getColor());
32 }
```

### 6.11.3.2 getCenter()

```
point obstacle::getCenter ( )
```

getter of the center point attribute

Definition at line 34 of file obstacle.cpp.

```
34     {
35     return p;
36 }
```

### 6.11.3.3 getRadius()

```
float obstacle::getRadius ( )
```

getter of the radius attribute

Definition at line 42 of file obstacle.cpp.

```
42     {
43     return r;
44 }
```

### 6.11.3.4 setCenter()

```
void obstacle::setCenter (
    point p )
```

setter of the center point attribute

Definition at line 38 of file obstacle.cpp.

```
38     {
39     this->p = p;
40 }
```

### 6.11.3.5 setRadius()

```
void obstacle::setRadius (
    float r )
```

setter of the radius attribute

Definition at line 46 of file obstacle.cpp.

```
46     {
47     this->r = r;
48 }
```

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

- /home/user/Desktop/mmm/autonomousSteeringAgents/include/[obstacle.h](#)
- /home/user/Desktop/mmm/autonomousSteeringAgents/src/[obstacle.cpp](#)

## 6.12 obstacleAvoidance Class Reference

```
#include <obstacleAvoidance.h>
```

### Public Member Functions

- [obstacleAvoidance](#) ()  
*default constructor.*

### Static Public Member Functions

- static void [loop](#) ()  
*obstacle avoidance scenario loop function*
- static void [createObstacle](#) (vector< [obstacle](#) > &[obstacles](#))  
*creation of list of obstacles*

### Static Public Attributes

- static vector< [obstacle](#) > [obstacles](#)  
*list of obstacles*

### Additional Inherited Members

#### 6.12.1 Detailed Description

Definition at line 15 of file obstacleAvoidance.h.

#### 6.12.2 Constructor & Destructor Documentation

### 6.12.2.1 obstacleAvoidance()

```
obstacleAvoidance::obstacleAvoidance ( )
```

default constructor.

Definition at line 43 of file obstacleAvoidance.cpp.

```
44 {
45     name = "avoid obstacles";
46     createStaticAgents("ag1", "ag2");
47     createObstacle(obstacles);
48     callback = reinterpret_cast <void(*)()> ( (void *)(&loop) );
49 }
```

## 6.12.3 Member Function Documentation

### 6.12.3.1 createObstacle()

```
void obstacleAvoidance::createObstacle (
    vector< obstacle > & obstacles ) [static]
```

creation of list of obstacles

Parameters

|                  |                    |
|------------------|--------------------|
| <i>obstacles</i> | list to be created |
|------------------|--------------------|

Note

opengl callback forces that function to be static

Definition at line 36 of file obstacleAvoidance.cpp.

```
37 {
38     obstacles.push_back(obstacle(point(0,0), 8));
39     obstacles.push_back(obstacle(point(-20,0), 3));
40     obstacles.push_back(obstacle(point(20,-10), 4));
41 }
```

### 6.12.3.2 loop()

```
void obstacleAvoidance::loop ( ) [static]
```

obstacle avoidance scenario loop function



**Note**

opengl callback forces that function to be static

Definition at line 17 of file obstacleAvoidance.cpp.

```

18 {
19     for(auto it = agents.begin(); it < agents.end(); it++){
20         (*it).setTarget(view.getMousePosition());
21         pvector seek = behavior.seek(*it);
22         seek.mul(0.5);
23
24         pvector avoid = behavior.avoid(obstacles, *it);
25         (*it).force = avoid + seek;
26         (*it).arrive = true;
27     }
28
29     for(auto it = obstacles.begin(); it < obstacles.end(); it++){
30         (*it).draw(view);
31     }
32
33     refresh();
34 }
```

**6.12.4 Member Data Documentation****6.12.4.1 obstacles**

```
vector< obstacle > obstacleAvoidance::obstacles [static]
```

list of obstacles

**Note**

opengl callback forces that function to be static

Definition at line 32 of file obstacleAvoidance.h.

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

- /home/user/Desktop/mmm/autonomousSteeringAgents/include/[obstacleAvoidance.h](#)
- /home/user/Desktop/mmm/autonomousSteeringAgents/src/[obstacleAvoidance.cpp](#)

**6.13 path Class Reference**

```
#include <path.h>
```

## Public Member Functions

- [path](#) ()  
*default constructor.*
- [path](#) (float width)  
*donstructor.*
- void [addPoint](#) ([point](#) p)  
*adds a new point to the path*
- void [setPathWidth](#) (int w)  
*setter of the path width*
- int [getPathWidth](#) ()  
*getter of the path width*
- void [draw](#) ([graphics](#) view)  
*overriden draw implementation*

## Public Attributes

- vector< [point](#) > [points](#)  
*list of points added to the path*

### 6.13.1 Detailed Description

Definition at line 17 of file path.h.

### 6.13.2 Constructor & Destructor Documentation

#### 6.13.2.1 [path\(\)](#) [1/2]

```
path::path ( )
```

default constructor.

See also

[path\(float width\)](#)

Definition at line 16 of file path.cpp.

```
17 {  
18     setColor(BLUE);  
19     width = 8;  
20 }
```

#### 6.13.2.2 [path\(\)](#) [2/2]

```
path::path (  
    float width )
```

donstructor.

## Parameters

|              |                        |
|--------------|------------------------|
| <i>width</i> | The width of the path. |
|--------------|------------------------|

## See also

[path\(\)](#)

Definition at line 31 of file path.cpp.

```
32 {  
33     this->width = width;  
34     setColor(BLUE);  
35 }
```

### 6.13.3 Member Function Documentation

#### 6.13.3.1 addPoint()

```
void path::addPoint (  
    point p )
```

adds a new point to the path

## Parameters

|              |                    |
|--------------|--------------------|
| <i>point</i> | to add to the path |
|--------------|--------------------|

Definition at line 11 of file path.cpp.

```
12 {  
13     points.push_back(p);  
14 }
```

#### 6.13.3.2 draw()

```
void path::draw (  
    graphics view ) [virtual]
```

overriden draw implementation

Implements [entity](#).

Definition at line 37 of file path.cpp.

```
37     {  
38         view.drawPath(*this);  
39     }
```

### 6.13.3.3 getPathWidth()

```
int path::getPathWidth ( )
```

getter of the path width

Definition at line 26 of file path.cpp.

```
26         {  
27     return width;  
28 }
```

### 6.13.3.4 setPathWidth()

```
void path::setPathWidth (  
    int w )
```

setter of the path width

Definition at line 22 of file path.cpp.

```
22     {  
23     width = w;  
24 }
```

## 6.13.4 Member Data Documentation

### 6.13.4.1 points

```
vector<point> path::points
```

list of points added to the path

Definition at line 41 of file path.h.

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

- [/home/user/Desktop/mmm/autonomousSteeringAgents/include/path.h](#)
- [/home/user/Desktop/mmm/autonomousSteeringAgents/src/path.cpp](#)

## 6.14 pathFollower Class Reference

```
#include <pathFollower.h>
```

### Public Member Functions

- [pathFollower \(\)](#)  
*default constructor.*

## Static Public Member Functions

- static void `loop` ()  
*path follower scenario loop function*
- static void `createPath` (`path` &p)  
*creates path*

## Static Public Attributes

- static `path` `myPath`  
*path that will be followed*

## Additional Inherited Members

### 6.14.1 Detailed Description

Definition at line 14 of file pathFollower.h.

### 6.14.2 Constructor & Destructor Documentation

#### 6.14.2.1 pathFollower()

```
pathFollower::pathFollower ( )
```

default constructor.

Definition at line 38 of file pathFollower.cpp.

```
39 {  
40     int agentCount = 40;  
41     float maxForce = 0.2;  
42     float maxSpeed = 0.4;  
43     myPath = path(8);  
44     createPath(myPath);  
45     name = "path following";  
46     createRandomAgents(agentCount, maxForce, maxSpeed);  
47     callback = reinterpret_cast<void(*)()> ( (void *)(&loop) );  
48 }
```

### 6.14.3 Member Function Documentation

#### 6.14.3.1 createPath()

```
void pathFollower::createPath (  
    path & p ) [static]
```

creates path

## Parameters

|             |           |
|-------------|-----------|
| <i>path</i> | to create |
|-------------|-----------|

## Note

opengl callback forces that function to be static

Definition at line 30 of file pathFollower.cpp.

```

31 {
32     p.addPoint(point(-40, 5));
33     p.addPoint(point(-14, 15));
34     p.addPoint(point( 10, 7));
35     p.addPoint(point( 40, 12));
36 }
```

### 6.14.3.2 loop()

```
void pathFollower::loop ( ) [static]
```

path follower scenario loop function

## Note

opengl callback forces that function to be static

Definition at line 17 of file pathFollower.cpp.

```

18 {
19     for(auto it = agents.begin(); it < agents.end(); it++){
20         pvector flwpth = behavior.stayInPath(*it, myPath, view);
21         pvector sep = behavior.separation(agents, *it, 3);
22         sep.mul(5);
23         (*it).force = sep + flwpth;
24     }
25     myPath.draw(view);
26 }
27 refresh();
28 }
```

## 6.14.4 Member Data Documentation

### 6.14.4.1 myPath

```
path pathFollower::myPath [static]
```

path that will be followed

## Note

opengl callback forces that function to be static

Definition at line 38 of file pathFollower.h.

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

- /home/user/Desktop/mmm/autonomousSteeringAgents/include/[pathFollower.h](#)
- /home/user/Desktop/mmm/autonomousSteeringAgents/src/[pathFollower.cpp](#)

## 6.15 point Class Reference

```
#include <point.h>
```

### Public Member Functions

- `point ()`  
*default constructor*
- `point (float x, float y)`  
*constructor*
- `void div (float d)`  
*divide point*
- `void mul (float d)`  
*multiply point*
- `void print (const string &s)`  
*debug function*
- `void getNormalPoint (point predicted, point start, point end)`  
*provides normal point on a vector of a point*
- `point operator+ (pvector const &obj)`  
*overloaded + operator*
- `point operator+ (point const &obj)`  
*overloaded + operator*
- `pvector operator- (point const &obj)`  
*overloaded - operator*
- `bool operator== (point const &obj)`  
*overloaded == operator*

### Public Attributes

- `float x`  
*x position*
- `float y`  
*y position*

#### 6.15.1 Detailed Description

Definition at line 15 of file point.h.

#### 6.15.2 Constructor & Destructor Documentation

### 6.15.2.1 point() [1/2]

```
point::point ( )
```

default constructor

See also

[point\(float x, float y\)](#)

Definition at line 21 of file point.cpp.

```
22 {  
23     x = 0;  
24     y = 0;  
25 }
```

### 6.15.2.2 point() [2/2]

```
point::point (  
                float x,  
                float y )
```

constructor

Parameters

|          |                         |
|----------|-------------------------|
| <i>x</i> | position x of the point |
| <i>y</i> | position y of the point |

See also

[point\(\)](#)

Definition at line 15 of file point.cpp.

```
16 {  
17     this->x = x;  
18     this->y = y;  
19 }
```

## 6.15.3 Member Function Documentation

### 6.15.3.1 div()

```
void point::div (  
                float d )
```

divide point



## Parameters

|          |  |
|----------|--|
| <i>d</i> | scalar to divide position of the point |
|----------|--|

Definition at line 42 of file point.cpp.

```
43 {
44     x = x / d;
45     y = y / d;
46 }
```

## 6.15.3.2 getNormalPoint()

```
void point::getNormalPoint (
    point predicted,
    point start,
    point end )
```

provides normal point on a vector of a point

## Parameters

|                  |  |
|------------------|--|
| <i>predicted</i> | point that caller require normal on the vector |
| <i>start</i>     | point of the vector                            |
| <i>end</i>       | point of the vector                            |

Definition at line 71 of file point.cpp.

```
72 {
73     pvector a = predicted - start;
74     pvector b = end - start;
75     b.normalize();
76     float a_dot_b = a.dotProduct(b);
77     b.mul(a_dot_b);
78     point normalPoint = start + b;
79     this->x = normalPoint.x;
80     this->y = normalPoint.y;
81 }
```

## 6.15.3.3 mul()

```
void point::mul (
    float d )
```

multiply point

## Parameters

|          |  |
|----------|--|
| <i>d</i> | scalar to multiply position of the point |
|----------|--|

Definition at line 48 of file point.cpp.

```
49 {  
50     x = x * d;  
51     y = y * d;  
52 }
```

#### 6.15.3.4 operator+() [1/2]

```
point point::operator+ (  
    point const & obj )
```

overloaded + operator

##### Parameters

|            |              |
|------------|--------------|
| <i>obj</i> | point to add |
|------------|--------------|

##### Returns

sum

Definition at line 55 of file point.cpp.

```
56 {  
57     point res;  
58     res.x = x + obj.x;  
59     res.y = y + obj.y;  
60     return res;  
61 }
```

#### 6.15.3.5 operator+() [2/2]

```
point point::operator+ (  
    pvector const & obj )
```

overloaded + operator

##### Parameters

|            |               |
|------------|---------------|
| <i>obj</i> | vector to add |
|------------|---------------|

##### Returns

sum

Definition at line 27 of file point.cpp.

```
28 {  
29     point res;  
30     res.x = x + obj.x;  
31     res.y = y + obj.y;  
32     return res;  
33 }
```

### 6.15.3.6 operator-()

```
pvector point::operator- (
    point const & obj )
```

overloaded - operator

#### Parameters

|            |                   |
|------------|-------------------|
| <i>obj</i> | point to subtract |
|------------|-------------------|

#### Returns

difference

Definition at line 63 of file point.cpp.

```
64 {
65     pvector res;
66     res.x = x - obj.x;
67     res.y = y - obj.y;
68     return res;
69 }
```

### 6.15.3.7 operator==()

```
bool point::operator== (
    point const & obj )
```

overloaded == operator

#### Parameters

|            |                  |
|------------|------------------|
| <i>obj</i> | point to compare |
|------------|------------------|

#### Returns

comparison result

Definition at line 35 of file point.cpp.

```
36 {
37     if(x == obj.x && y == obj.y)
38         return true;
39     return false;
40 }
```

### 6.15.3.8 print()

```
void point::print (
    const string & s )
```

debug function

#### Parameters

|   |                               |
|---|-------------------------------|
| s | explanation string of the log |
|---|-------------------------------|

Definition at line 83 of file point.cpp.

```
84 {  
85     cout << " " << s << " " << x << " " << y << endl;  
86 }
```

### 6.15.4 Member Data Documentation

#### 6.15.4.1 x

```
float point::x
```

x position

Definition at line 88 of file point.h.

#### 6.15.4.2 y

```
float point::y
```

y position

Definition at line 93 of file point.h.

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

- [/home/user/Desktop/mmm/autonomousSteeringAgents/include/point.h](#)
- [/home/user/Desktop/mmm/autonomousSteeringAgents/src/point.cpp](#)

## 6.16 prison Class Reference

```
#include <prison.h>
```

### Public Member Functions

- [prison](#) ()  
*default constructor.*

## Static Public Member Functions

- static void [loop](#) ()  
*prisoning scenario loop function*

## Additional Inherited Members

### 6.16.1 Detailed Description

Definition at line 15 of file prison.h.

### 6.16.2 Constructor & Destructor Documentation

#### 6.16.2.1 prison()

```
prison::prison ( )
```

default constructor.

Definition at line 31 of file prison.cpp.

```
32 {
33     int agentCount = 10;
34     float maxForce = 0.6;
35     float maxSpeed = 0.6;
36
37     name = "stay in prison";
38     createRandomAgents(agentCount, maxForce, maxSpeed);
39     callback = reinterpret_cast<void(*)()> ( (void *)(&loop) );
40 }
```

### 6.16.3 Member Function Documentation

#### 6.16.3.1 loop()

```
void prison::loop ( ) [static]
```

prisoning scenario loop function

prison loop function

#### Note

opengl callback forces that function to be static

Definition at line 18 of file prison.cpp.

```
19 {
20     for(auto it = agents.begin(); it < agents.end(); it++){
21         view.drawLine(point(-WALL, WALL), point(WALL, WALL), BLUE);
22         view.drawLine(point(WALL, WALL), point(WALL, -WALL), BLUE);
23         view.drawLine(point(WALL, -WALL), point(-WALL, -WALL), BLUE);
24         view.drawLine(point(-WALL, WALL), point(-WALL, -WALL), BLUE);
25         (*it).force = behavior.stayInArea(*it, WALL - DISTANCE);
26         (*it).force += behavior.separation(agents, *it, 4);
27     }
28     refresh();
29 }
```

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

- /home/user/Desktop/mmm/autonomousSteeringAgents/include/[prison.h](#)
- /home/user/Desktop/mmm/autonomousSteeringAgents/src/[prison.cpp](#)

## 6.17 pursuit Class Reference

```
#include <pursuit.h>
```

### Public Member Functions

- [pursuit](#) ()  
*default constructor.*

### Static Public Member Functions

- static void [loop](#) ()  
*pursuing scenario loop function*

### Additional Inherited Members

#### 6.17.1 Detailed Description

Definition at line 14 of file pursuit.h.

#### 6.17.2 Constructor & Destructor Documentation

##### 6.17.2.1 pursuit()

```
pursuit::pursuit ( )
```

default constructor.

Definition at line 31 of file pursuit.cpp.

```
32 {  
33     name = "pursuit";  
34     createStaticAgents("gazelle", "lion");  
35     callback = reinterpret_cast <void(*)()> ( (void *)(&loop) );  
36 }
```

#### 6.17.3 Member Function Documentation

### 6.17.3.1 loop()

```
void pursuit::loop ( ) [static]
```

pursuing scenario loop function

#### Note

opengl callback forces that function to be static

Definition at line 15 of file pursuit.cpp.

```
16 {
17     for(auto it = agents.begin(); it < agents.end(); it++){
18         if((*it).getName() == "gazelle"){
19             (*it).setTarget(view.getMousePosition());
20             (*it).force = behavior.seek(*it);
21         }
22         else{//lion
23             (*it).force = behavior.pursuit(agents, *it, view, "gazelle");
24         }
25         (*it).arrive = true;
26     }
27     refresh();
28 }
29 }
```

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

- /home/user/Desktop/mmm/autonomousSteeringAgents/include/pursuit.h
- /home/user/Desktop/mmm/autonomousSteeringAgents/src/pursuit.cpp

## 6.18 pvector Class Reference

```
#include <pvector.h>
```

### Public Member Functions

- [pvector \(\)](#)  
*default constructor*
- [pvector \(float x, float y\)](#)  
*constructor*
- float [magnitude \(\)](#)  
*calculates magnitude of the vector*
- [pvector & normalize \(\)](#)  
*normalize*
- void [div \(float i\)](#)  
*vector division*
- void [mul \(float i\)](#)  
*vector multiplication*
- void [add \(pvector p\)](#)  
*addition of vectors*
- void [limit \(float limit\)](#)  
*vector limitation*
- float [getAngle \(\)](#)

- calculates vector angle*
- float `dotProduct` (`pvector` v)  
*dot product of two vectors*
- float `angleBetween` (`pvector` v)  
*angle calculation between two vectors*
- void `print` (const string &s)  
*debug function*
- `pvector operator+=` (`pvector` const &obj)  
*overloaded += operator*
- `pvector operator+` (`pvector` const &obj)  
*overloaded + operator*
- `pvector operator-` (`pvector` const &obj)  
*overloaded - operator*
- `pvector operator-` (`point` const &obj)  
*overloaded - operator*
- `pvector operator+` (`point` const &obj)  
*overloaded + operator*
- bool `operator==` (`pvector` const &obj)  
*overloaded == operator*

## Public Attributes

- float `x`  
*x magnitude of the vector*
- float `y`  
*y magnitude of the vector*

### 6.18.1 Detailed Description

Definition at line 17 of file `pvector.h`.

### 6.18.2 Constructor & Destructor Documentation

#### 6.18.2.1 `pvector()` [1/2]

```
pvector::pvector ( )
```

default constructor

See also

`pvector(float x, float y)`

Definition at line 35 of file `pvector.cpp`.

```
36 {
37     x = 0;
38     y = 0;
39 }
```



### 6.18.2.2 pvector() [2/2]

```
pvector::pvector (
    float x,
    float y )
```

constructor

#### Parameters

|          |                         |
|----------|-------------------------|
| <i>x</i> | magnitude of the vector |
| <i>y</i> | magnitude of the vector |

See also

[pvector\(\)](#)

Definition at line 41 of file pvector.cpp.

```
42 {
43     this->x = x;
44     this->y = y;
45 }
```

## 6.18.3 Member Function Documentation

### 6.18.3.1 add()

```
void pvector::add (
    pvector p )
```

addition of vectors

#### Parameters

|          |               |
|----------|---------------|
| <i>p</i> | vector to add |
|----------|---------------|

Definition at line 59 of file pvector.cpp.

```
60 {
61     x = x + p.x;
62     y = y + p.y;
63 }
```

### 6.18.3.2 angleBetween()

```
float pvector::angleBetween (
    pvector v )
```

angle calculation between two vectors

**Parameters**

|          |                           |
|----------|---------------------------|
| <i>v</i> | vector to calculate angle |
|----------|---------------------------|

**Returns**

angle

Definition at line 23 of file pvector.cpp.

```
24 {  
25     float angle = this->dotProduct(v) / (this->magnitude() * v.magnitude());  
26     angle = acos(angle) * 180 / PI;  
27     return angle;  
28 }
```

**6.18.3.3 div()**

```
void pvector::div (  
    float i )
```

vector division

**Parameters**

|          |                        |
|----------|------------------------|
| <i>i</i> | scalar value to divide |
|----------|------------------------|

Definition at line 47 of file pvector.cpp.

```
48 {  
49     x = x / i;  
50     y = y / i;  
51 }
```

**6.18.3.4 dotProduct()**

```
float pvector::dotProduct (  
    pvector v )
```

dot product of two vectors

**Parameters**

|          |                                 |
|----------|---------------------------------|
| <i>v</i> | vector to calculate dot product |
|----------|---------------------------------|

**Returns**

returns scalar dot product

Definition at line 30 of file pvector.cpp.

```

31 {
32     return ((x * v.x) + (y * v.y));
33 }

```

### 6.18.3.5 getAngle()

```
float pvector::getAngle ( )
```

calculates vector angle

#### Returns

angle

Definition at line 16 of file pvector.cpp.

```

17 {
18     float angle;
19     angle = atan2 (this->y, this->x) * 180 / PI;
20     return angle;
21 }

```

### 6.18.3.6 limit()

```
void pvector::limit (
    float limit )
```

vector limitation

#### Parameters

|              |                                    |
|--------------|------------------------------------|
| <i>limit</i> | value to restrict vector magnitude |
|--------------|------------------------------------|

Definition at line 84 of file pvector.cpp.

```

85 {
86     this->normalize();
87     this->mul(limit);
88 }

```

### 6.18.3.7 magnitude()

```
float pvector::magnitude ( )
```

calculates magnitude of the vector

#### Returns

magnitude of the vector

Definition at line 65 of file pvector.cpp.

```

66 {
67     return sqrt((this->x * this->x) + (this->y * this->y));
68 }

```

### 6.18.3.8 mul()

```
void pvector::mul (
    float i )
```

vector multiplication

#### Parameters

|          |                          |
|----------|--------------------------|
| <i>i</i> | scalar value to multiply |
|----------|--------------------------|

Definition at line 53 of file pvector.cpp.

```
54 {
55     x = x * i;
56     y = y * i;
57 }
```

### 6.18.3.9 normalize()

```
pvector & pvector::normalize ( )
```

normalize

#### Returns

normalized vector

Definition at line 70 of file pvector.cpp.

```
71 {
72     float magnitude = this->magnitude();
73     if(magnitude != 0){
74         this->x = this->x / magnitude;
75         this->y = this->y / magnitude;
76     }
77     else{
78         this->x = 0;
79         this->y = 0;
80     }
81     return *this;
82 }
```

### 6.18.3.10 operator+() [1/2]

```
pvector pvector::operator+ (
    point const & obj )
```

overloaded + operator

#### Parameters

|            |              |
|------------|--------------|
| <i>obj</i> | point to add |
|------------|--------------|

**Returns**

sum

Definition at line 112 of file pvector.cpp.

```
113 {  
114     pvector res;  
115     res.x = x + obj.x;  
116     res.y = y + obj.y;  
117     return res;  
118 }
```

**6.18.3.11 operator+() [2/2]**

```
pvector pvector::operator+ (  
    pvector const & obj )
```

overloaded + operator

**Parameters**

|            |               |
|------------|---------------|
| <i>obj</i> | vector to add |
|------------|---------------|

**Returns**

sum

Definition at line 90 of file pvector.cpp.

```
91 {  
92     pvector res;  
93     res.x = x + obj.x;  
94     res.y = y + obj.y;  
95     return res;  
96 }
```

**6.18.3.12 operator+=()**

```
pvector pvector::operator+= (  
    pvector const & obj )
```

overloaded += operator

**Parameters**

|            |               |
|------------|---------------|
| <i>obj</i> | vector to add |
|------------|---------------|

**Returns**

sum

Definition at line 98 of file pvector.cpp.

```
99 {  
100     x = x + obj.x;  
101     y = y + obj.y;  
102     return *this;  
103 }
```

### 6.18.3.13 operator-() [1/2]

```
pvector pvector::operator- (  
    point const & obj )
```

overloaded - operator

#### Parameters

|            |                   |
|------------|-------------------|
| <i>obj</i> | point to subtract |
|------------|-------------------|

#### Returns

difference

Definition at line 120 of file pvector.cpp.

```
121 {  
122     pvector res;  
123     res.x = x - obj.x;  
124     res.y = y - obj.y;  
125     return res;  
126 }
```

### 6.18.3.14 operator-() [2/2]

```
pvector pvector::operator- (  
    pvector const & obj )
```

overloaded - operator

#### Parameters

|            |                    |
|------------|--------------------|
| <i>obj</i> | vector to subtract |
|------------|--------------------|

#### Returns

difference

Definition at line 133 of file pvector.cpp.

```
134 {  
135     pvector res;  
136     res.x = x - obj.x;  
137     res.y = y - obj.y;  
138     return res;  
139 }
```

### 6.18.3.15 operator==()

```
bool pvector::operator== (
    pvector const & obj )
```

overloaded == operator

#### Parameters

|            |                          |
|------------|--------------------------|
| <i>obj</i> | vector to check if equal |
|------------|--------------------------|

#### Returns

comparison result

Definition at line 105 of file pvector.cpp.

```
106 {
107     if (x == obj.x && y == obj.y)
108         return true;
109     return false;
110 }
```

### 6.18.3.16 print()

```
void pvector::print (
    const string & s )
```

debug function

#### Parameters

|          |                     |
|----------|---------------------|
| <i>s</i> | identification text |
|----------|---------------------|

Definition at line 128 of file pvector.cpp.

```
129 {
130     cout << s << " " << x << " " << y << endl;
131 }
```

## 6.18.4 Member Data Documentation

### 6.18.4.1 x

```
float pvector::x
```

x magnitude of the vector

Definition at line 140 of file pvector.h.

### 6.18.4.2 y

```
float pvector::y
```

y magnitude of the vector

Definition at line 145 of file pvector.h.

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

- [/home/user/Desktop/mmm/autonomousSteeringAgents/include/pvector.h](#)
- [/home/user/Desktop/mmm/autonomousSteeringAgents/src/pvector.cpp](#)

## 6.19 random Class Reference

```
#include <random.h>
```

### Static Public Member Functions

- static void [createRandomArray](#) (int \*arr, int size)  
*random array generation*

#### 6.19.1 Detailed Description

Definition at line 9 of file random.h.

#### 6.19.2 Member Function Documentation

##### 6.19.2.1 createRandomArray()

```
void random::createRandomArray (
    int * arr,
    int size ) [static]
```

random array generation

#### Parameters

|             |                                    |
|-------------|------------------------------------|
| <i>arr</i>  | struct that includes random values |
| <i>size</i> | of the array                       |

Definition at line 14 of file random.cpp.



```

14                                     {
15     srand(time(NULL));
16     for(int i=0; i<size; i++)
17         arr[i] = i+1;
18
19     for (int i=0; i < size; i++){
20         int r = rand() % size;
21         swap(arr[i], arr[r]);
22     }
23 }

```

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

- [/home/user/Desktop/mmm/autonomousSteeringAgents/include/random.h](#)
- [/home/user/Desktop/mmm/autonomousSteeringAgents/src/random.cpp](#)

## 6.20 scenario Class Reference

```
#include <scenario.h>
```

### Public Member Functions

- [scenario](#) ()  
*default constructor.*
- void [initGL](#) (int \*argv, char \*\*argc)  
*graphics initialization*
- void [createRandomAgents](#) (int agentCount, float mForce, float mSpeed)  
*random agent creation*
- void [createStaticAgents](#) (string s1, string s2)  
*static agent creation*
- void [createTroop](#) (int agentCount)  
*troop creation*

### Static Public Member Functions

- static void [refresh](#) ()  
*refreshes all items*

### Public Attributes

- void(\* [callback](#) )()  
*openGL screen refresh callback function, used as main loop in derived classes*

### Static Public Attributes

- static vector< [agent](#) > [agents](#)  
*structure stores agents*
- static [graphics view](#)  
*graphics instance used*
- static [steeringBehavior behavior](#)  
*behavior instance used*
- static string [name](#)  
*scenario name*

### 6.20.1 Detailed Description

Definition at line 19 of file scenario.h.

### 6.20.2 Constructor & Destructor Documentation

#### 6.20.2.1 scenario()

```
scenario::scenario ( )
```

default constructor.

Definition at line 28 of file scenario.cpp.

```
29 {
30     view = graphics();
31 }
```

### 6.20.3 Member Function Documentation

#### 6.20.3.1 createRandomAgents()

```
void scenario::createRandomAgents (
    int agentCount,
    float mForce,
    float mSpeed )
```

random agent creation

##### Parameters

|                   |                  |
|-------------------|------------------|
| <i>agentCount</i> | number of agents |
| <i>mForce</i>     | maxForce         |
| <i>mSpeed</i>     | maxSpeed         |

Definition at line 45 of file scenario.cpp.

```
46 {
47     int size = MAX_NUMBER_OF_AGENTS * 2;
48     int arr[size];
49
50     srand(time(NULL));
51     random::createRandomArray(arr, size);
52     agent tempAgent {0, 0};
53
54     for(int i=0; i < count * 2; i=i+2){
55         tempAgent.setName("agent "+to_string(i)+":");
56         tempAgent.position.x = arr[i] - WIDTH;
57         tempAgent.position.y = arr[i+1] - HEIGHT;
58         tempAgent.setColor(color::getColor((i/2) % 8));
59         tempAgent.setFeatures(speed, force, 5, 1);
60         agents.push_back(tempAgent);
```

```

61     }
62 }

```

### 6.20.3.2 createStaticAgents()

```

void scenario::createStaticAgents (
    string s1,
    string s2 )

```

static agent creation

#### Parameters

|           |                          |
|-----------|--------------------------|
| <i>s1</i> | name of the first agent  |
| <i>s2</i> | name of the second agent |

Definition at line 64 of file scenario.cpp.

```

65 {
66     agent agent1 {-10.0, 0.0};
67     agent1.id = 1;
68     agent1.setName(s1);
69     agent1.setColor(BLUE);
70     agent1.setFeatures(0.5, 0.2, 5, 1);
71     agents.push_back(agent1);
72
73     agent agent2 { 10.0, 0.0};
74     agent2.id = 2;
75     agent2.setName(s2);
76     agent2.setColor(RED);
77     agent2.setFeatures(0.4, 0.2, 5, 1);
78     agents.push_back(agent2);
79 }

```

### 6.20.3.3 createTroop()

```

void scenario::createTroop (
    int agentCount )

```

troop creation

#### Parameters

|               |                        |
|---------------|------------------------|
| <i>number</i> | of agents in the troop |
|---------------|------------------------|

Definition at line 81 of file scenario.cpp.

```

82 {
83     int row = 14;
84     int blanks = 5;
85     int rowStartPosition = -33;
86     agent tempAgent {0, 0};
87     pvector location {-33, 33};
88
89     for(int i=0; i < count; i++){
90         tempAgent.id = i;
91         tempAgent.setVelocity(pvector(0, 0));
92         tempAgent.position.x = location.x;

```

```

93     tempAgent.position.y = location.y;
94     tempAgent.setTarget(tempAgent.position);
95
96     if( ((i+1) % row) == 0){
97         location.y -= blanks;
98         location.x = rowStartPosition;
99     }
100    else
101        location.x += blanks;
102
103    tempAgent.setColor(color::getColor((i/2) % 8));
104    tempAgent.setFeatures(0.3, 0.3, 5, 1);
105    agents.push_back(tempAgent);
106 }
107 }

```

#### 6.20.3.4 initGL()

```

void scenario::initGL (
    int * argv,
    char ** argc )

```

graphics initialization

##### Parameters

|             |                          |
|-------------|--------------------------|
| <i>argv</i> | list of user arguments   |
| <i>argc</i> | number of user arguments |

Definition at line 23 of file scenario.cpp.

```

24 {
25     view.initGraphics(argc, argv, callback);
26 }

```

#### 6.20.3.5 refresh()

```

void scenario::refresh ( ) [static]

```

refreshes all items

##### Note

opengl callback forces that function to be static

Definition at line 33 of file scenario.cpp.

```

34 {
35     point textPosition = point(-34, 32.25);
36
37     for(auto it = agents.begin(); it < agents.end(); it++){
38         (*it).draw(view);
39     }
40
41     view.drawText(name, textPosition);
42     view.refreshScene();
43 }

```

## 6.20.4 Member Data Documentation

### 6.20.4.1 agents

```
vector< agent > scenario::agents [static]
```

structure stores agents

#### Note

opengl callback forces that function to be static

Definition at line 43 of file scenario.h.

### 6.20.4.2 behavior

```
steeringBehavior scenario::behavior [static]
```

behavior instance used

#### Note

opengl callback forces that function to be static

Definition at line 55 of file scenario.h.

### 6.20.4.3 callback

```
void(* scenario::callback) ()
```

openGL screen refresh callback function, used as main loop in derived classes

Definition at line 66 of file scenario.h.

### 6.20.4.4 name

```
string scenario::name [static]
```

scenario name

#### Note

opengl callback forces that function to be static

Definition at line 61 of file scenario.h.

#### 6.20.4.5 view

`graphics` `scenario::view` [static]

graphics instance used

##### Note

opengl callback forces that function to be static

Definition at line 49 of file `scenario.h`.

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

- `/home/user/Desktop/mmm/autonomousSteeringAgents/include/scenario.h`
- `/home/user/Desktop/mmm/autonomousSteeringAgents/src/scenario.cpp`

## 6.21 steeringBehavior Class Reference

```
#include <steeringBehavior.h>
```

### Public Member Functions

- `pvector stayInArea` (`agent &agent`, int turnPoint)  
*gets reflection force*
- `pvector inFlowField` (`agent &agent`, `flowField` &flow)  
*gets flow field force*
- `pvector stayInPath` (`agent &agent`, `path` &path, `graphics` view)  
*gets force to follow path*
- `pvector seek` (`agent &agent`)  
*force to seek*
- `pvector separation` (vector< `agent` > agents, `agent &agent`, float radius)  
*force to separate*
- `pvector cohesion` (vector< `agent` > boids, `agent &agent`, float radius)  
*force to cohesion*
- `pvector align` (vector< `agent` > boids, `agent &agent`, float radius)  
*force to align*
- `pvector wander` (`agent &agent`)  
*force to wander*
- `pvector pursuit` (vector< `agent` > boids, `agent` &pursuer, `graphics` view, string name)  
*force to pursue*
- `pvector evade` (vector< `agent` > boids, `agent` &evader, `graphics` view, string name)  
*force to evade*
- `pvector flee` (`agent &agent`, `graphics` &view, `point` p)  
*force to flee*
- `pvector avoid` (vector< `obstacle` > obstacles, `agent &agent`)  
*force to avoid*

### 6.21.1 Detailed Description

Definition at line 36 of file steeringBehavior.h.

### 6.21.2 Member Function Documentation

#### 6.21.2.1 align()

```
pvector steeringBehavior::align (
    vector< agent > boids,
    agent & agent,
    float radius )
```

force to align

##### Parameters

|               |                                       |
|---------------|---------------------------------------|
| <i>agent</i>  | to be aligned                         |
| <i>boids</i>  | list of all the agents                |
| <i>radius</i> | range for agents that will be aligned |

##### Returns

force to be applied

Definition at line 120 of file steeringBehavior.cpp.

```
121 {
122     pvector sum {0,0};
123     int count = 0;
124     for(auto it = boids.begin(); it < boids.end(); it++){
125         float d = (agent.position - (*it).position).magnitude();
126         if( (d > 0) && (d < radius) ){
127             sum += (*it).getVelocity();
128             count++;
129         }
130     }
131     if(count > 0){
132         sum.div(count);
133         sum.normalize().mul(agent.maxSpeed);
134         agent.steering = sum - agent.getVelocity();
135         return agent.steering;
136     }
137     return pvector(0,0);
138 }
```

#### 6.21.2.2 avoid()

```
pvector steeringBehavior::avoid (
    vector< obstacle > obstacles,
    agent & agent )
```

force to avoid

## Parameters

|                  |                                      |
|------------------|--------------------------------------|
| <i>agent</i>     | agent that will avoid from obstacles |
| <i>obstacles</i> | list of all existing objects         |

## Returns

force to be applied

Definition at line 181 of file steeringBehavior.cpp.

```

182 {
183     float dynamic_length = agent.getVelocity().magnitude() / agent.maxSpeed;
184     pvector vel = agent.getVelocity();
185     vel.normalize().mul(dynamic_length);
186     pvector ahead = vel + agent.position;
187     vel.mul(6);
188     pvector ahead2 = vel + agent.position;
189     //view.drawPoint(point(ahead.x, ahead.y));
190     //view.drawPoint(point(ahead2.x, ahead2.y));
191
192     for(auto it = obstacles.begin(); it < obstacles.end(); it++){
193         float dist = (ahead - (*it).getCenter()).magnitude();
194         float dist2 = (ahead2 - (*it).getCenter()).magnitude();
195         if(dist < (*it).getRadius() + 2 || dist2 < (*it).getRadius() + 2){
196             pvector avoidance = ahead - (*it).getCenter();
197             avoidance.normalize().mul(20);
198             /*a = point(avoidance.x, avoidance.y);
199             view.drawLine(agent.position, agent.position + a, color(0,1,0));*/
200             return avoidance;
201         }
202     }
203     return pvector(0,0);
204 }
```

### 6.21.2.3 cohesion()

```

pvector steeringBehavior::cohesion (
    vector< agent > boids,
    agent & agent,
    float radius )
```

force to cohesion

## Parameters

|               |  |
|---------------|--|
| <i>agent</i>  | to go to center of other agents, with specified distance |
| <i>boids</i>  | list of all the agents                                   |
| <i>radius</i> | range for agents that will be aligned                    |

## Returns

force to be applied

Definition at line 140 of file steeringBehavior.cpp.

```

141 {
142     point sum {0,0};
143     int count = 0;
```



```

144     for(auto it = boids.begin(); it < boids.end(); it++){
145         float d = (agent.position - (*it).position).magnitude();
146         if( (d > 0) && (d < radius) ){
147             sum = sum + (*it).position;
148             count++;
149         }
150     }
151     if(count > 0){
152         sum.div(count);
153         agent.setTarget(sum);
154         return seek(agent);
155     }
156     return pvector(0,0);
157 }

```

### 6.21.2.4 evade()

```

pvector steeringBehavior::evade (
    vector< agent > boids,
    agent & evader,
    graphics view,
    string name )

```

force to evade

#### Parameters

|               |                        |
|---------------|------------------------|
| <i>evader</i> | agent that will escape |
| <i>view</i>   | used for debugging     |
| <i>boids</i>  | list of all the agents |
| <i>name</i>   | other agent to evade   |

#### Returns

force to be applied

Definition at line 47 of file steeringBehavior.cpp.

```

48 {
49     agent target;
50     for(auto it = boids.begin(); it < boids.end(); it++){
51         if((*it).getName() == name){
52             target = *it;
53         }
54     }
55
56     point p = point(evader.position.x + 2, evader.position.y - 2);
57     view.drawText(evader.getName(), p);
58     p = point(target.position.x + 2, target.position.y - 2);
59     view.drawText(target.getName(), p);
60
61     pvector targetVel = target.getVelocity();
62     targetVel.mul(5); //TODO: magic number
63
64     point futurePos = target.position + targetVel;
65     //view.drawPoint(futurePos);
66
67     pvector dist = evader.position - futurePos;
68     dist.normalize().mul( 1 / dist.magnitude() );
69
70     evader.setTarget(evader.position + dist);
71     return flee(evader, view, futurePos);
72 }

```

### 6.21.2.5 flee()

```
pvector steeringBehavior::flee (
    agent & agent,
    graphics & view,
    point p )
```

force to flee

#### Parameters

|              |                        |
|--------------|------------------------|
| <i>agent</i> | agent that will flee   |
| <i>view</i>  | used for debugging     |
| <i>p</i>     | point that agent flees |

#### Returns

force to be applied

Definition at line 28 of file steeringBehavior.cpp.

```
29 {
30     int radius = 15;
31
32     pvector dist = agent.getTarget() - p;
33     //view.drawPoint(agent.targetPoint);
34
35     if(dist.magnitude() < radius){
36         agent.arrive = false;
37         agent.desiredVelocity = agent.position - p;
38     }
39     else{
40         agent.arrive = true;
41         agent.desiredVelocity = agent.getTarget() - agent.position;
42     }
43     agent.steering = agent.desiredVelocity - agent.getVelocity();
44     return agent.steering;
45 }
```

### 6.21.2.6 inFlowField()

```
pvector steeringBehavior::inFlowField (
    agent & agent,
    flowField & flow )
```

gets flow field force

#### Parameters

|              |                          |
|--------------|--------------------------|
| <i>agent</i> | unit to apply flow field |
| <i>flow</i>  | field                    |

#### Returns

force to be applied

Definition at line 236 of file steeringBehavior.cpp.

```

237 {
238     //pos_x, pos_y must be non negative integer
239     int pos_x = abs((int)agent.position.x) % WIDTH;
240     int pos_y = abs((int)agent.position.y) % HEIGHT;
241     //TODO: modification required for non uniform fields
242     return flow.getField(pos_x, pos_y);
243 }

```

### 6.21.2.7 pursuit()

```

pvector steeringBehavior::pursuit (
    vector< agent > boids,
    agent & pursuer,
    graphics view,
    string name )

```

force to pursue

#### Parameters

|                |  |
|----------------|--|
| <i>pursuer</i> | agent that will follow specified agent |
| <i>view</i>    | used for debugging                     |
| <i>boids</i>   | list of all the agents                 |
| <i>name</i>    | other agent to pursue                  |

#### Returns

force to be applied

Definition at line 74 of file steeringBehavior.cpp.

```

75 {
76     agent target;
77     for(auto it = boids.begin(); it < boids.end(); it++){
78         if((*it).getName() == name){
79             target = *it;
80         }
81     }
82
83     point p = point(target.position.x + 2, target.position.y - 2);
84     view.drawText(target.getName(), p);
85
86     p = point(pursuer.position.x + 2, pursuer.position.y - 2);
87     view.drawText(pursuer.getName(), p);
88
89     float dist = (target.position - pursuer.position).magnitude();
90     float t = dist / target.maxSpeed;
91
92     pvector targetVel = target.getVelocity();
93     targetVel.mul(t);
94     point futurePos = target.position + targetVel;
95     pursuer.setTarget(futurePos);
96     return seek(pursuer);
97 }

```

### 6.21.2.8 seek()

```

pvector steeringBehavior::seek (
    agent & agent )

```

force to seek

## Parameters

|              |                                       |
|--------------|---------------------------------------|
| <i>agent</i> | that will go to specific target point |
|--------------|---------------------------------------|

## Returns

force to be applied

Definition at line 206 of file steeringBehavior.cpp.

```

207 {
208     agent.desiredVelocity = agent.getTarget() - agent.position;
209     agent.steering = agent.desiredVelocity - agent.getVelocity();
210     return agent.steering;
211 }
```

### 6.21.2.9 separation()

```

pvector steeringBehavior::separation (
    vector< agent > agents,
    agent & agent,
    float radius )
```

force to separate

## Parameters

|               |                                       |
|---------------|---------------------------------------|
| <i>agent</i>  | agent that will be stayed away        |
| <i>agents</i> | list of all the agents                |
| <i>radius</i> | range for agents that will be aligned |

## Returns

force to be applied

Definition at line 159 of file steeringBehavior.cpp.

```

160 {
161     pvector sum = pvector(0,0);
162     int count = 0;
163     for(auto it = agents.begin(); it < agents.end(); it++){
164         float d = (agent.position - (*it).position).magnitude();
165         if( (d > 0) && (d < radius) ){
166             pvector diff = agent.position - (*it).position;
167             diff.normalize().div(d);
168             sum = sum + diff;
169             count++;
170         }
171     }
172     if(count > 0){
173         sum.div(count);
174         sum.normalize().mul(agent.maxSpeed);
175         agent.steering = sum - agent.getVelocity();
176         return agent.steering;
177     }
178     return pvector(0,0);
179 }
```

### 6.21.2.10 stayInArea()

```
pvector steeringBehavior::stayInArea (
    agent & agent,
    int turnPoint )
```

gets reflection force

#### Parameters

|                  |                               |
|------------------|-------------------------------|
| <i>agent</i>     | unit to check                 |
| <i>turnpoint</i> | defines border to apply force |

#### Returns

force to be applied

Definition at line 245 of file steeringBehavior.cpp.

```
246 {
247     if(agent.position.x >= turnPoint){
248         agent.desiredVelocity = pvector( -agent.maxSpeed, agent.getVelocity().y );
249         agent.steering = agent.desiredVelocity - agent.getVelocity();
250         return agent.steering;
251     }
252     else if(agent.position.x <= -turnPoint){
253         agent.desiredVelocity = pvector( agent.maxSpeed, agent.getVelocity().y );
254         agent.steering = agent.desiredVelocity - agent.getVelocity();
255         return agent.steering;
256     }
257     else if(agent.position.y >= turnPoint){
258         agent.desiredVelocity = pvector( agent.getVelocity().x, -agent.maxSpeed );
259         agent.steering = agent.desiredVelocity - agent.getVelocity();
260         return agent.steering;
261     }
262     else if(agent.position.y <= -turnPoint){
263         agent.desiredVelocity = pvector( agent.getVelocity().x, agent.maxSpeed );
264         agent.steering = agent.desiredVelocity - agent.getVelocity();
265         return agent.steering;
266     }
267     return pvector(0,0);
268 }
```

### 6.21.2.11 stayInPath()

```
pvector steeringBehavior::stayInPath (
    agent & agent,
    path & path,
    graphics view )
```

gets force to follow path

#### Parameters

|              |                     |
|--------------|---------------------|
| <i>agent</i> | to follow the pathk |
| <i>path</i>  | to follow           |
| <i>view</i>  | used for debugging  |

**Returns**

force to be applied

Definition at line 213 of file steeringBehavior.cpp.

```

214 {
215     float worldRecord = 1000000;
216     point normalPoint, predictedPos, start, end;
217     pvector distance;
218     for(auto it = path.points.begin(); it < path.points.end()-1; it++){
219         start = point ((*it).x, (*it).y);
220         end = point ((*it+1).x, (*it+1).y);
221         predictedPos = agent.position + agent.getVelocity();
222         normalPoint.getNormalPoint(predictedPos, start, end);
223         if (normalPoint.x < start.x || normalPoint.x > end.x){
224             normalPoint = end;
225         }
226         distance = predictedPos - normalPoint;
227         if (distance.magnitude() < worldRecord){
228             worldRecord = distance.magnitude();
229             agent.setTarget(end);
230         }
231         view.drawPoint(agent.getTarget());
232     }
233     return seek(agent);
234 }
```

**6.21.2.12 wander()**

```
pvector steeringBehavior::wander (
    agent & agent )
```

force to wander

**Parameters**

|              |                        |
|--------------|------------------------|
| <i>agent</i> | agent that will wander |
|--------------|------------------------|

**Returns**

force to be applied

Definition at line 99 of file steeringBehavior.cpp.

```

100 {
101     pvector circleCenter = agent.getVelocity();
102     circleCenter.normalize().mul(CIRCLE_DISTANCE + CIRCLE_RADIUS);
103
104     int wanderAngle = (rand() % 360);
105     pvector displacement {0, 1};
106     setAngle(displacement, wanderAngle);
107     displacement.mul(CIRCLE_RADIUS);
108
109     agent.desiredVelocity = displacement + circleCenter;
110     agent.steering = agent.desiredVelocity - agent.getVelocity();
111
112     //move it to the center when it is out of screen
113     if(agent.position.x > WIDTH || agent.position.x < -WIDTH ||
114        agent.position.y > HEIGHT || agent.position.y < -HEIGHT)
115         agent.position = point(0,0);
116
117     return agent.steering;
118 }
```

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

- /home/user/Desktop/mmm/autonomousSteeringAgents/include/steeringBehavior.h
- /home/user/Desktop/mmm/autonomousSteeringAgents/src/steeringBehavior.cpp

## 6.22 wander Class Reference

```
#include <wander.h>
```

### Public Member Functions

- `wander ()`  
*default constructor*

### Static Public Member Functions

- static void `loop ()`  
*wander scenario loop function*

### Additional Inherited Members

#### 6.22.1 Detailed Description

Definition at line 14 of file wander.h.

#### 6.22.2 Constructor & Destructor Documentation

##### 6.22.2.1 wander()

```
wander::wander ( )
```

default constructor

**Todo** business logic will be changed

Definition at line 23 of file wander.cpp.

```
24 {  
25     int agentCount = 30;  
26     float maxForce = 0.3;  
27     float maxSpeed = 0.6;  
28     name = "wandering objects";  
29     createRandomAgents(agentCount, maxForce, maxSpeed);  
30     callback = reinterpret_cast <void(*)()> ( (void *)(&loop) );  
31 }
```

##### 6.22.3 Member Function Documentation

### 6.22.3.1 loop()

```
void wander::loop ( ) [static]
```

wander scenario loop function

#### Note

opengl callback forces that function to be static

Definition at line 15 of file wander.cpp.

```
16 {  
17     for(auto it = agents.begin(); it < agents.end(); it++){  
18         (*it).force = behavior.wander(*it);  
19     }  
20     refresh();  
21 }
```

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

- /home/user/Desktop/mmm/autonomousSteeringAgents/include/wander.h
- /home/user/Desktop/mmm/autonomousSteeringAgents/src/wander.cpp

## 6.23 windy Class Reference

```
#include <windy.h>
```

### Public Member Functions

- [windy \(\)](#)  
*default constructor.*

### Static Public Member Functions

- static void [loop \(\)](#)  
*windy scenario loop function*

### Static Public Attributes

- static [flowField flow](#)  
*flow field used*

### Additional Inherited Members

#### 6.23.1 Detailed Description

Definition at line 15 of file windy.h.



## 6.23.2 Constructor & Destructor Documentation

### 6.23.2.1 windy()

windy::windy ( )

default constructor.

Definition at line 29 of file windy.cpp.

```
30 {
31     int agentCount = 50;
32     float maxForce = 0.3;
33     float maxSpeed = 0.6;
34
35     name = "flow field";
36     createRandomAgents(agentCount, maxForce,
37     maxSpeed);
38     callback = reinterpret_cast <void(*)()> ( (void *)(&loop) );
39 }
```

## 6.23.3 Member Function Documentation

### 6.23.3.1 loop()

void windy::loop ( ) [static]

windy scenario loop function

**Note**

opengl callback forces that function to be static

Definition at line 17 of file windy.cpp.

```
18 {
19     for(auto it = agents.begin(); it < agents.end(); it++){
20         flow = flowField(pvector(GRAVITY));
21         (*it).force = behavior.inFlowField(*it, flow);
22
23         flow = flowField(pvector(WIND_WEST));
24         (*it).force += behavior.inFlowField(*it, flow);
25     }
26     refresh();
27 }
```

## 6.23.4 Member Data Documentation

### 6.23.4.1 flow

flowField windy::flow [static]

flow field used

**Note**

opengl callback forces that function to be static

Definition at line 32 of file windy.h.

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

- /home/user/Desktop/mmm/autonomousSteeringAgents/include/windy.h
- /home/user/Desktop/mmm/autonomousSteeringAgents/src/windy.cpp



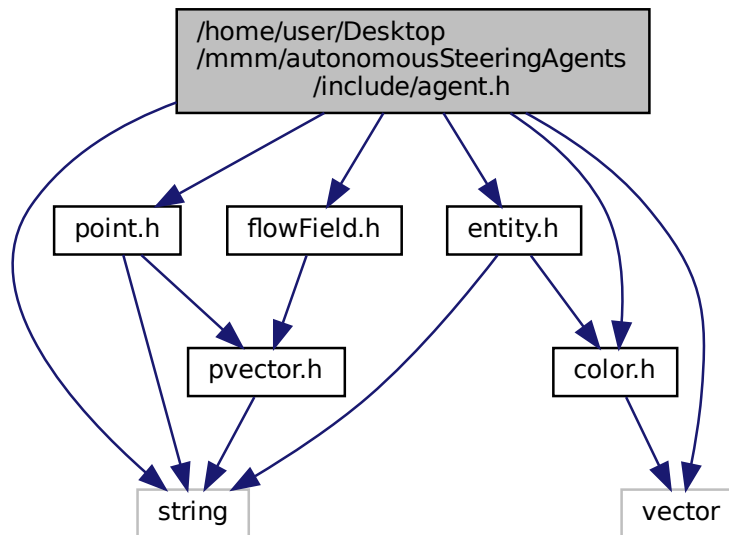
## Chapter 7

### File Documentation

- 7.1 /home/user/Desktop/mmm/autonomousSteeringAgents/agent.d File Reference**
- 7.2 /home/user/Desktop/mmm/autonomousSteeringAgents/color.d File Reference**
- 7.3 /home/user/Desktop/mmm/autonomousSteeringAgents/entity.d File Reference**
- 7.4 /home/user/Desktop/mmm/autonomousSteeringAgents/evade.d File Reference**
- 7.5 /home/user/Desktop/mmm/autonomousSteeringAgents/flee.d File Reference**
- 7.6 /home/user/Desktop/mmm/autonomousSteeringAgents/flock.d File Reference**
- 7.7 /home/user/Desktop/mmm/autonomousSteeringAgents/flowField.d File Reference**
- 7.8 /home/user/Desktop/mmm/autonomousSteeringAgents/graphics.d File Reference**
- 7.9 /home/user/Desktop/mmm/autonomousSteeringAgents/include/agent.h File Reference**

agent class defines all agent specifications

```
#include "point.h"
#include "color.h"
#include "entity.h"
#include "flowField.h"
#include <vector>
#include <string>
Include dependency graph for agent.h:
```



This graph shows which files directly or indirectly include this file:



## Classes

- class [agent](#)

### 7.9.1 Detailed Description

agent class defines all agent specifications

Author

Mehmet Rıza Öz - [mehmetrizaoz@gmail.com](mailto:mehmetrizaoz@gmail.com)

Date

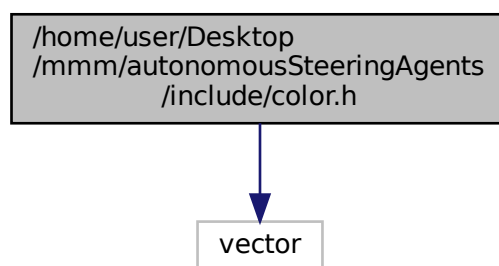
14.05.2021

## 7.10 /home/user/Desktop/mmm/autonomousSteeringAgents/include/color.h File Reference

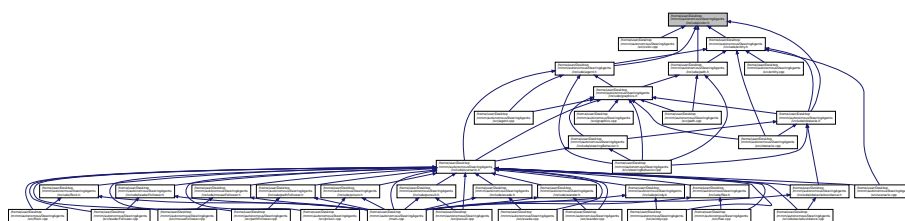
color class used for agent, path, wall etc. color

```
#include <vector>
```

Include dependency graph for color.h:



This graph shows which files directly or indirectly include this file:



### Classes

- class `color`

### Macros

- `#define BLACK color(0,0,0)`
- `#define BLUE color(0,0,1)`
- `#define GREEN color(0,1,0)`
- `#define CYAN color(0,1,1)`
- `#define RED color(1,0,0)`
- `#define YELLOW color(1,1,0)`
- `#define MAGENDA color(1,0,1)`
- `#define WHITE color(1,1,1)`

### 7.10.1 Detailed Description

color class used for agent, path, wall etc. color

#### Author

Mehmet Rıza Öz - [mehmetrizaoz@gmail.com](mailto:mehmetrizaoz@gmail.com)

#### Date

13.05.2021

### 7.10.2 Macro Definition Documentation

#### 7.10.2.1 BLACK

```
#define BLACK color(0,0,0)
```

Definition at line 10 of file color.h.

#### 7.10.2.2 BLUE

```
#define BLUE color(0,0,1)
```

Definition at line 11 of file color.h.

#### 7.10.2.3 CYAN

```
#define CYAN color(0,1,1)
```

Definition at line 13 of file color.h.

#### 7.10.2.4 GREEN

```
#define GREEN color(0,1,0)
```

Definition at line 12 of file color.h.

### 7.10.2.5 MAGENDA

```
#define MAGENDA color(1,0,1)
```

Definition at line 16 of file color.h.

### 7.10.2.6 RED

```
#define RED color(1,0,0)
```

Definition at line 14 of file color.h.

### 7.10.2.7 WHITE

```
#define WHITE color(1,1,1)
```

Definition at line 17 of file color.h.

### 7.10.2.8 YELLOW

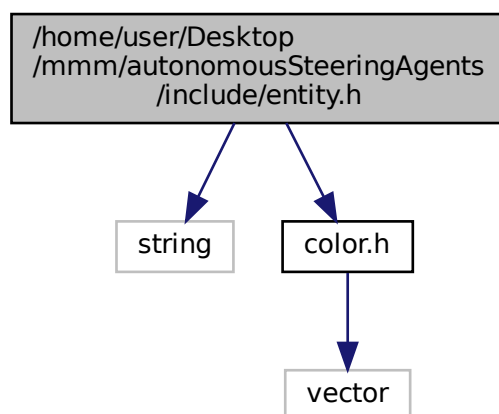
```
#define YELLOW color(1,1,0)
```

Definition at line 15 of file color.h.

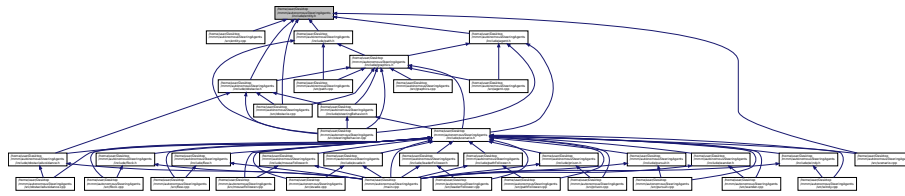
## 7.11 /home/user/Desktop/mmm/autonomousSteeringAgents/include/entity.h File Reference

```
#include <string>
#include "color.h"
```

Include dependency graph for entity.h:



This graph shows which files directly or indirectly include this file:



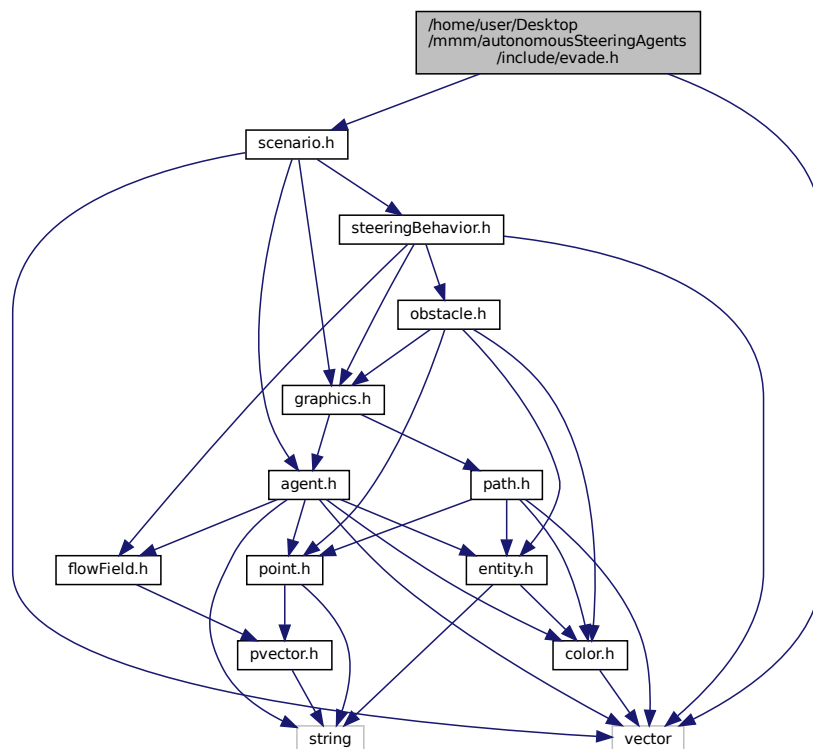
## Classes

- class [entity](#)

## 7.12 /home/user/Desktop/mmm/autonomousSteeringAgents/include/evade.h File Reference

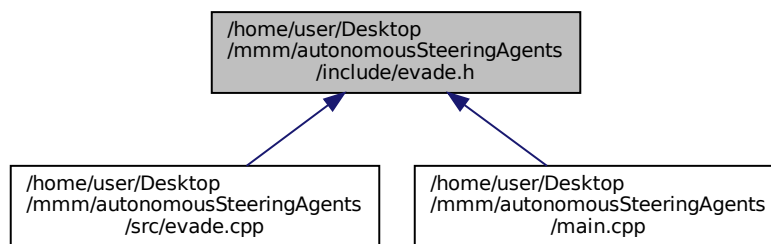
evade class inherited from scenario class

```
#include "scenario.h"
#include <vector>
Include dependency graph for evade.h:
```





This graph shows which files directly or indirectly include this file:



## Classes

- class [evade](#)

### 7.12.1 Detailed Description

evade class inherited from scenario class

#### Author

Mehmet Rıza Öz - [mehmetrizaoz@gmail.com](mailto:mehmetrizaoz@gmail.com)

#### Date

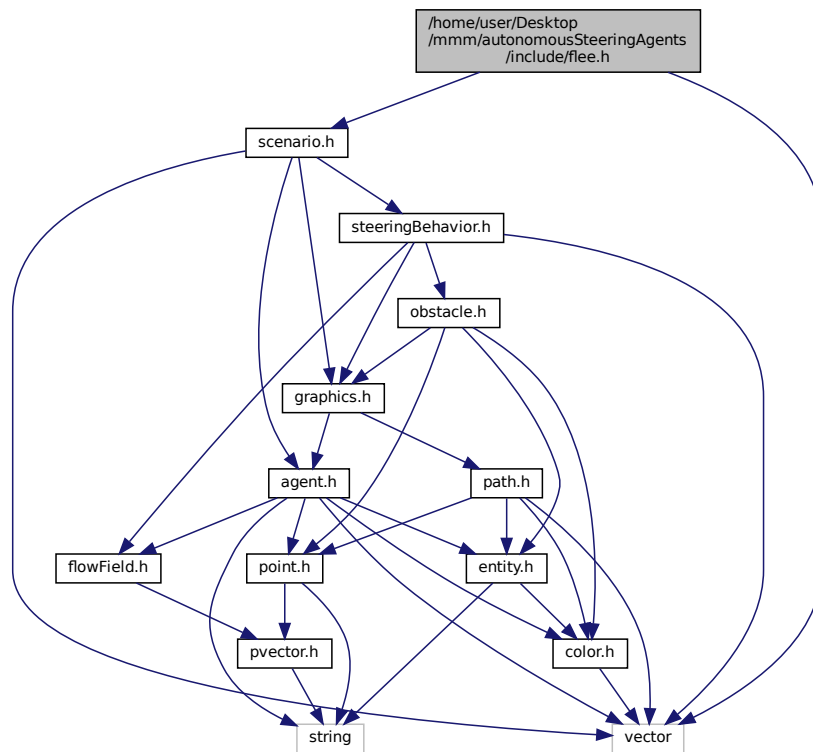
15.05.2021

## 7.13 /home/user/Desktop/mmm/autonomousSteeringAgents/include/flee.h File Reference

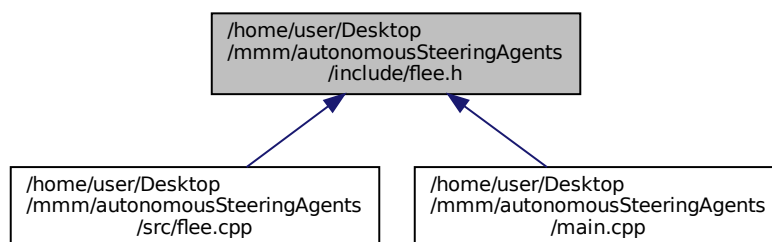
agents flee from mouse scenario

```
#include "scenario.h"
#include <vector>
```

Include dependency graph for flee.h:



This graph shows which files directly or indirectly include this file:



## Classes

- class [flee](#)

### 7.13.1 Detailed Description

agents flee from mouse scenario

## Author

Mehmet Rıza Öz - [mehmetrizaoz@gmail.com](mailto:mehmetrizaoz@gmail.com)

## Date

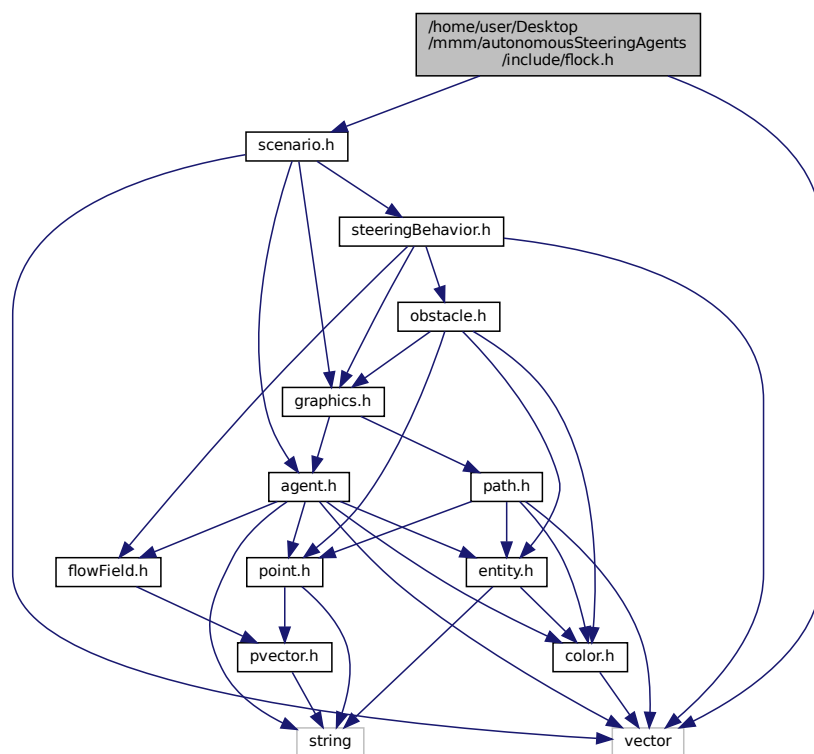
15.05.2021

## 7.14 /home/user/Desktop/mmm/autonomousSteeringAgents/include/flock.h File Reference

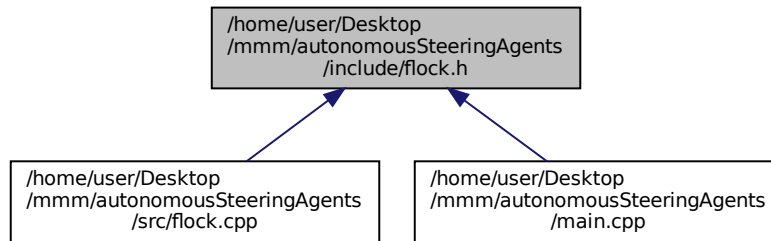
flocking agents scenario

```
#include "scenario.h"
#include <vector>
```

Include dependency graph for flock.h:



This graph shows which files directly or indirectly include this file:



## Classes

- class [flock](#)

### 7.14.1 Detailed Description

flocking agents scenario

#### Author

Mehmet Rıza Öz - [mehmetrizaoz@gmail.com](mailto:mehmetrizaoz@gmail.com)

#### Date

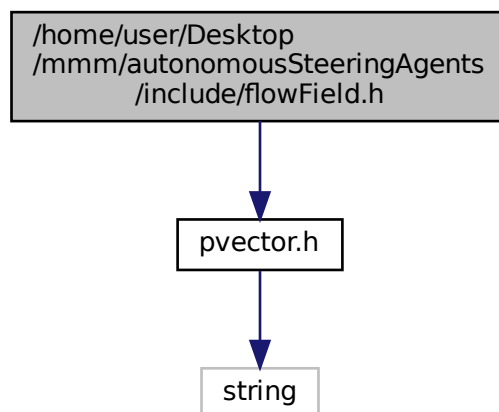
15.05.2021

## 7.15 [/home/user/Desktop/mmm/autonomousSteeringAgents/include/flowField.h](#) File Reference

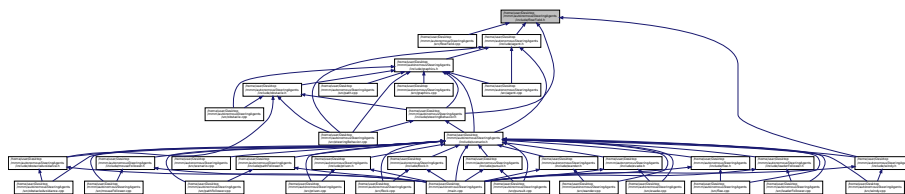
[flowField](#) class, screen can be filled with a force for each pixel

```
#include "pvector.h"
```

Include dependency graph for flowField.h:



This graph shows which files directly or indirectly include this file:



## Classes

- class [flowField](#)

## Macros

- `#define` [FIELD\\_WIDTH](#) 34
- `#define` [FIELD\\_HEIGHT](#) 34
- `#define` [WIND\\_WEST](#) 0.1, 0.0
- `#define` [GRAVITY](#) 0.0, -0.1

### 7.15.1 Detailed Description

[flowField](#) class, screen can be filled with a force for each pixel

#### Author

Mehmet Rıza Öz - [mehmetrizaoz@gmail.com](mailto:mehmetrizaoz@gmail.com)

#### Date

13.05.2021

## 7.15.2 Macro Definition Documentation

### 7.15.2.1 FIELD\_HEIGHT

```
#define FIELD_HEIGHT 34
```

Definition at line 13 of file flowField.h.

### 7.15.2.2 FIELD\_WIDTH

```
#define FIELD_WIDTH 34
```

Definition at line 12 of file flowField.h.

### 7.15.2.3 GRAVITY

```
#define GRAVITY 0.0, -0.1
```

Definition at line 16 of file flowField.h.

### 7.15.2.4 WIND\_WEST

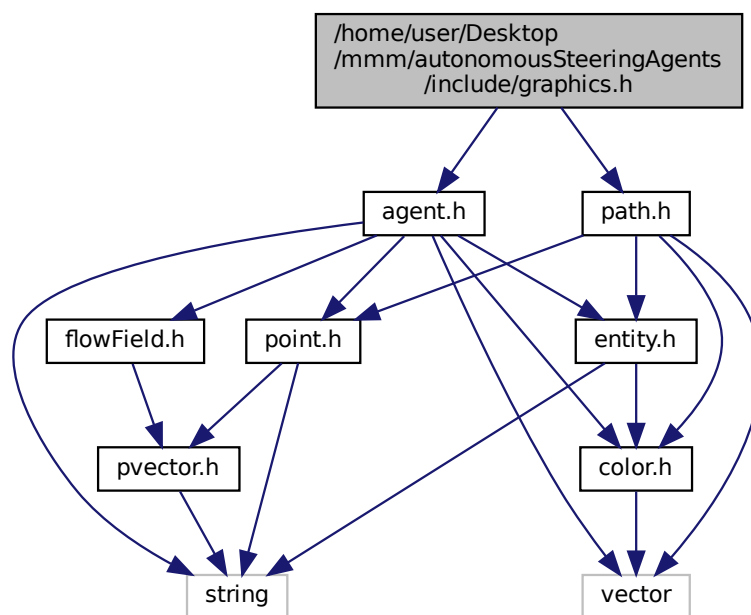
```
#define WIND_WEST 0.1, 0.0
```

Definition at line 15 of file flowField.h.

## 7.16 /home/user/Desktop/mmm/autonomousSteeringAgents/include/graphics.h File Reference

graphics class, drives OpenGL

```
#include "agent.h"
#include "path.h"
Include dependency graph for graphics.h:
```



This graph shows which files directly or indirectly include this file:



### Classes

- class [graphics](#)

### Macros

- `#define WIDTH 34`
- `#define HEIGHT 34`
- `#define ESC 27`
- `#define PI 3.14159265`

### 7.16.1 Detailed Description

graphics class, drives openGL

#### Author

Mehmet Rıza Öz - [mehmetrizaoz@gmail.com](mailto:mehmetrizaoz@gmail.com)

#### Date

15.05.2021

### 7.16.2 Macro Definition Documentation

#### 7.16.2.1 ESC

```
#define ESC 27
```

Definition at line 16 of file graphics.h.

#### 7.16.2.2 HEIGHT

```
#define HEIGHT 34
```

Definition at line 14 of file graphics.h.

#### 7.16.2.3 PI

```
#define PI 3.14159265
```

Definition at line 17 of file graphics.h.

#### 7.16.2.4 WIDTH

```
#define WIDTH 34
```

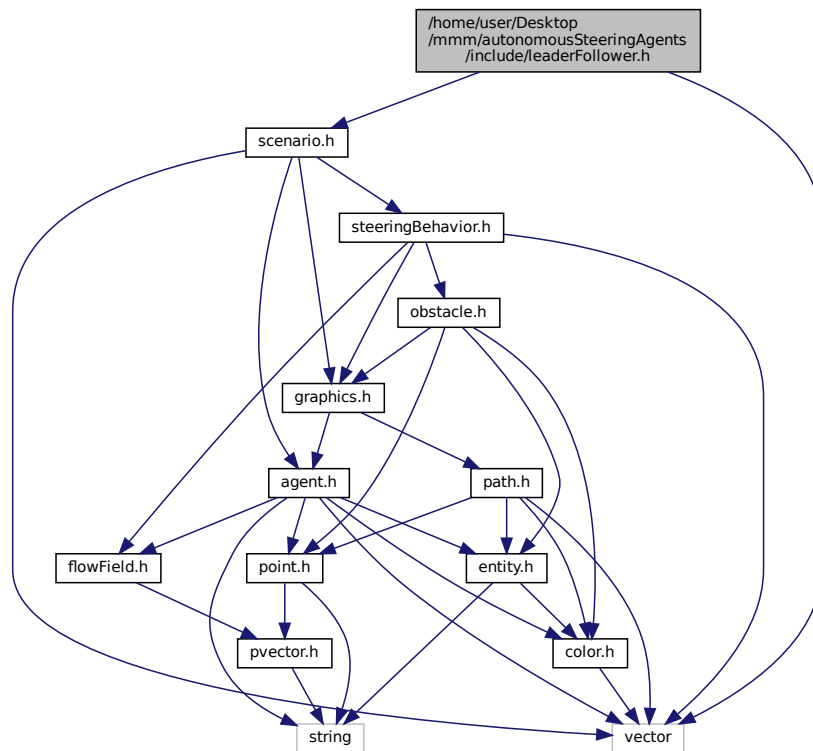
Definition at line 13 of file graphics.h.



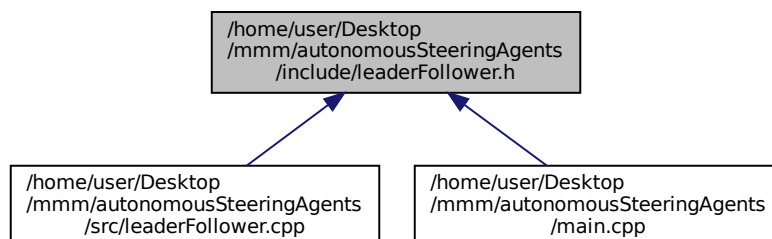
## 7.17 /home/user/Desktop/mmm/autonomousSteeringAgents/include/leaderFollower.h File Reference

agents follow leader scenario

```
#include "scenario.h"
#include <vector>
Include dependency graph for leaderFollower.h:
```



This graph shows which files directly or indirectly include this file:



## Classes

- class [leaderFollower](#)

### 7.17.1 Detailed Description

agents follow leader scenario

Author

Mehmet Rıza Öz - [mehmetrizaoz@gmail.com](mailto:mehmetrizaoz@gmail.com)

Date

19.05.2021

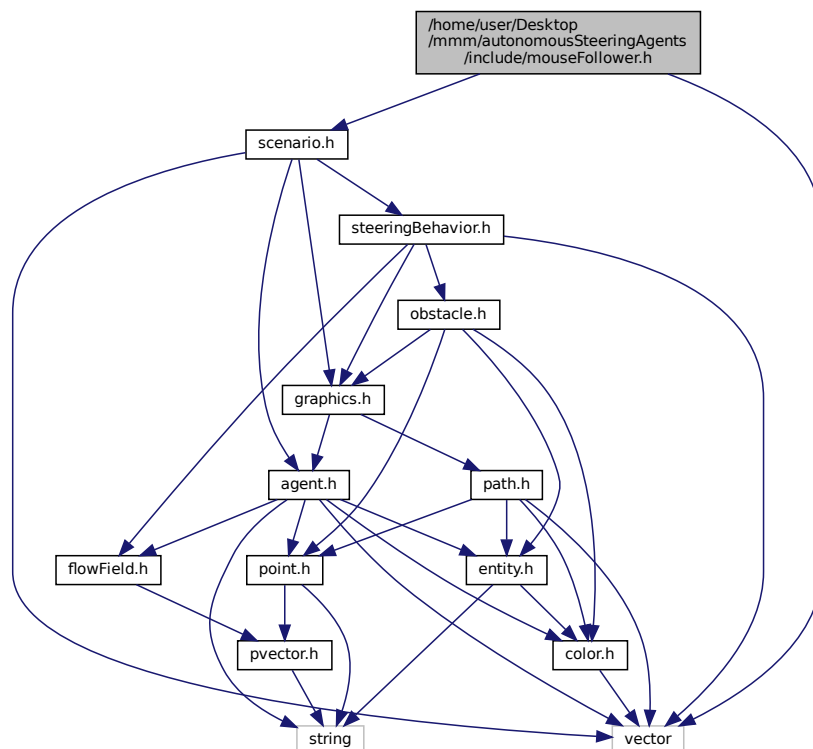
## 7.18 /home/user/Desktop/mmm/autonomousSteeringAgents/include/mouseFollower.h File Reference

agents follow mouse scenario

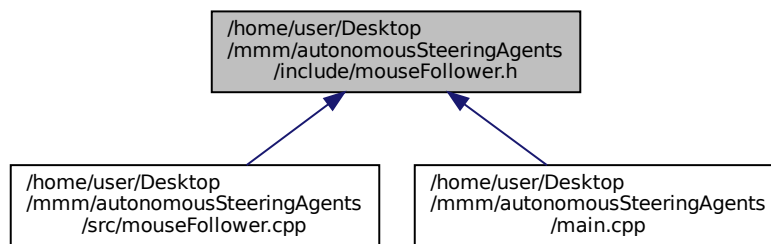
```
#include "scenario.h"
```

```
#include <vector>
```

Include dependency graph for mouseFollower.h:



This graph shows which files directly or indirectly include this file:



## Classes

- class [mouseFollower](#)

### 7.18.1 Detailed Description

agents follow mouse scenario

#### Author

Mehmet Rıza Öz - [mehmetrizaoz@gmail.com](mailto:mehmetrizaoz@gmail.com)

#### Date

15.05.2021

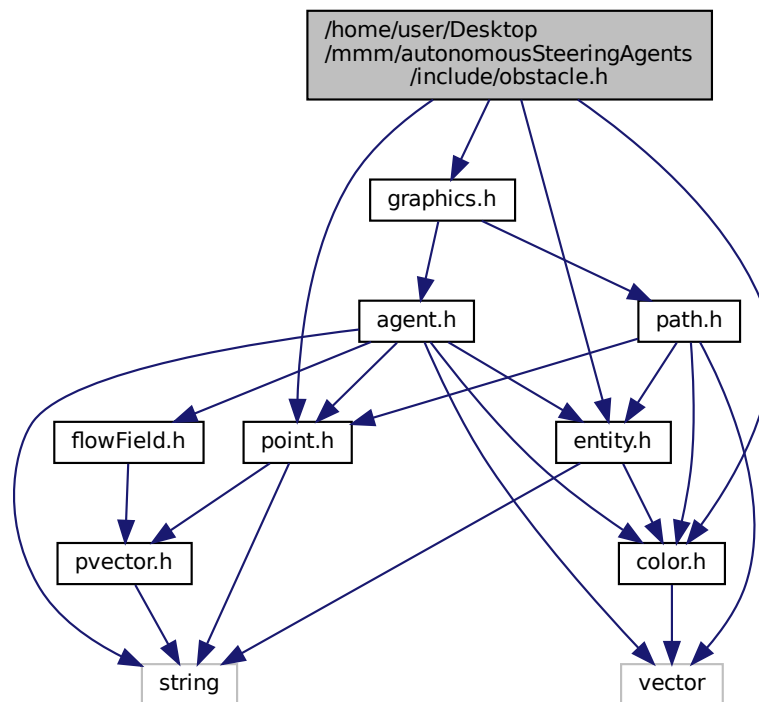
## 7.19 /home/user/Desktop/mmm/autonomousSteeringAgents/include/obstacle.h File Reference

circular obstacles for agent avoidance behaviors

```
#include "point.h"
#include "graphics.h"
#include "color.h"
```

```
#include "entity.h"
```

Include dependency graph for obstacle.h:



This graph shows which files directly or indirectly include this file:



## Classes

- class [obstacle](#)

### 7.19.1 Detailed Description

circular obstacles for agent avoidance behaviors

Author

Mehmet Rıza Öz - [mehmetrizaoz@gmail.com](mailto:mehmetrizaoz@gmail.com)

Date

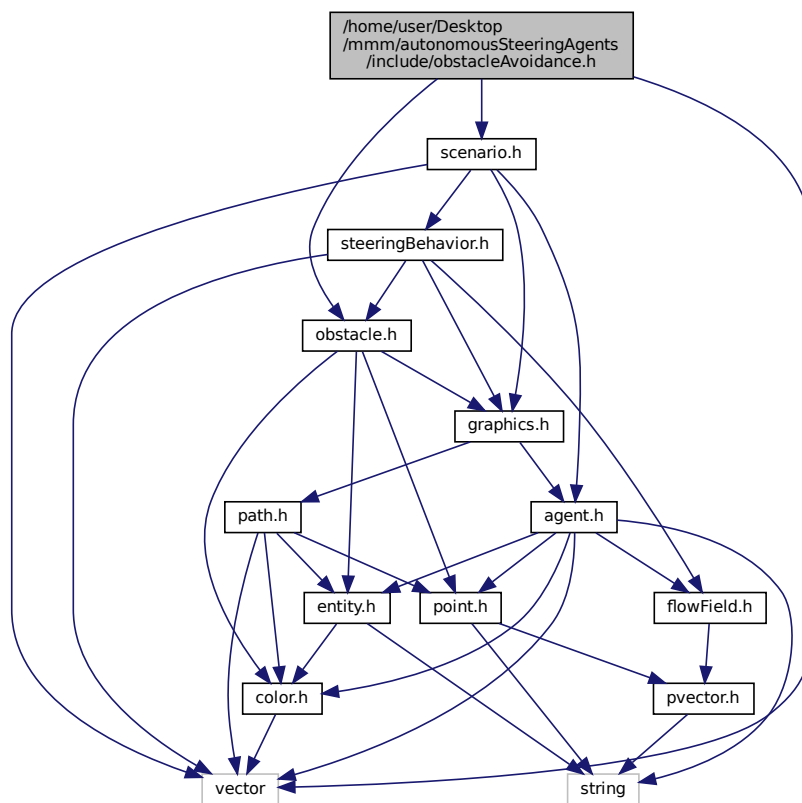
12.05.2021

## 7.20 /home/user/Desktop/mmm/autonomousSteeringAgents/include/obstacleAvoidance.h File Reference

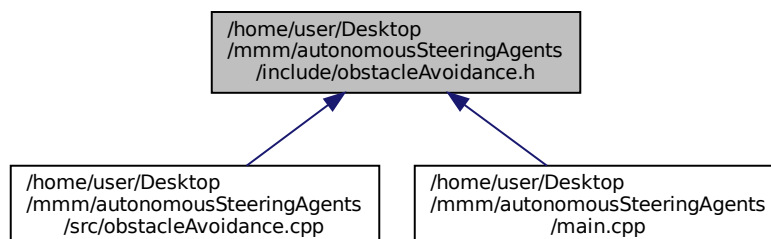
agents avoid from obstacles scenario

```
#include "scenario.h"
#include "obstacle.h"
#include <vector>
```

Include dependency graph for obstacleAvoidance.h:



This graph shows which files directly or indirectly include this file:



## Classes

- class [obstacleAvoidance](#)

### 7.20.1 Detailed Description

agents avoid from obstacles scenario

#### Author

Mehmet Rıza Öz - [mehmetrizaoz@gmail.com](mailto:mehmetrizaoz@gmail.com)

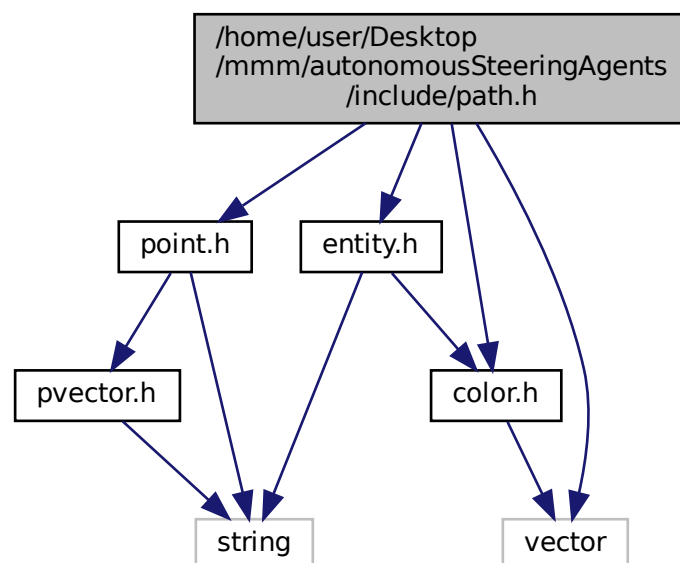
#### Date

15.05.2021

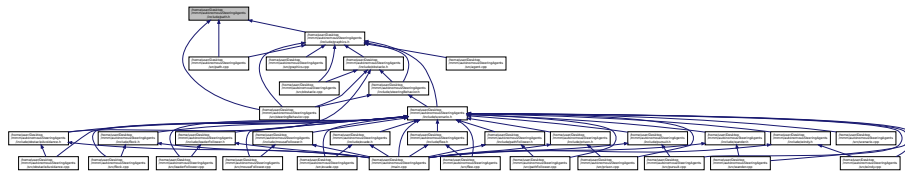
### 7.21 [/home/user/Desktop/mmm/autonomousSteeringAgents/include/path.h](#) File Reference

path class used for path following steering behaviors.

```
#include "point.h"
#include "entity.h"
#include "color.h"
#include <vector>
Include dependency graph for path.h:
```



This graph shows which files directly or indirectly include this file:



## Classes

- class [path](#)

### 7.21.1 Detailed Description

path class used for path following steering behaviors.

#### Author

Mehmet Rıza Öz - [mehmetrizaoz@gmail.com](mailto:mehmetrizaoz@gmail.com)

#### Date

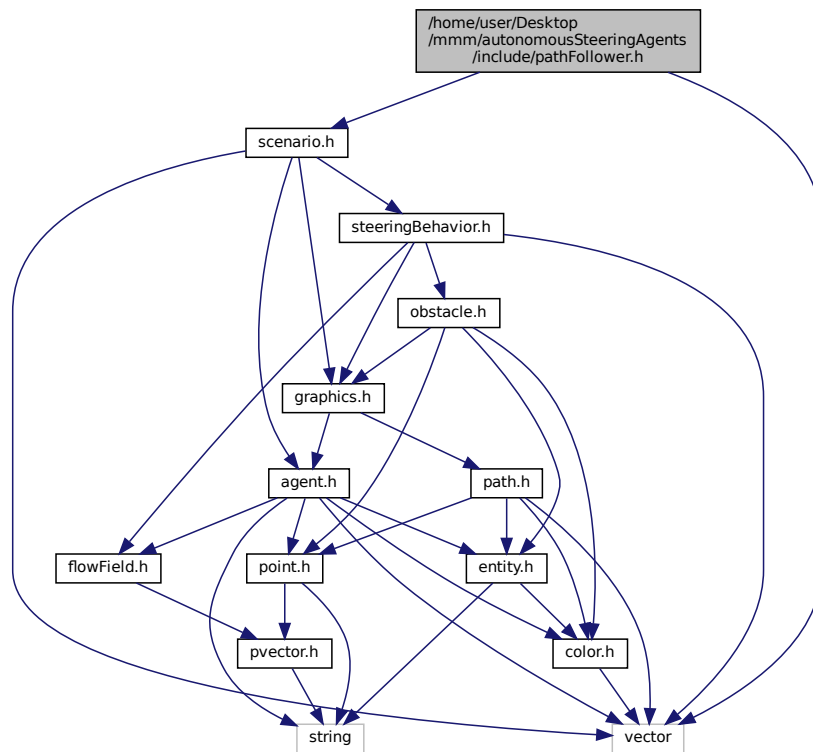
12.05.2021

## 7.22 /home/user/Desktop/mmm/autonomousSteeringAgents/include/pathFollower.h File Reference

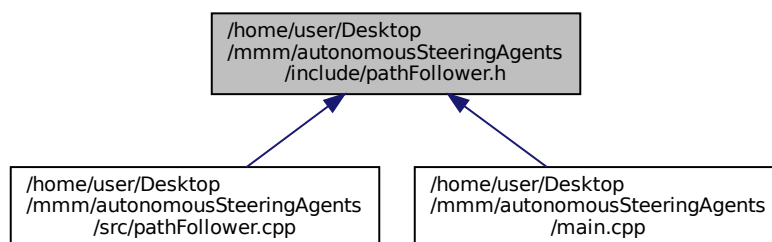
path following scenario

```
#include "scenario.h"
#include <vector>
```

Include dependency graph for pathFollower.h:



This graph shows which files directly or indirectly include this file:



## Classes

- class [pathFollower](#)

### 7.22.1 Detailed Description

path following scenario



## Author

Mehmet Rıza Öz - [mehmetrizaoz@gmail.com](mailto:mehmetrizaoz@gmail.com)

## Date

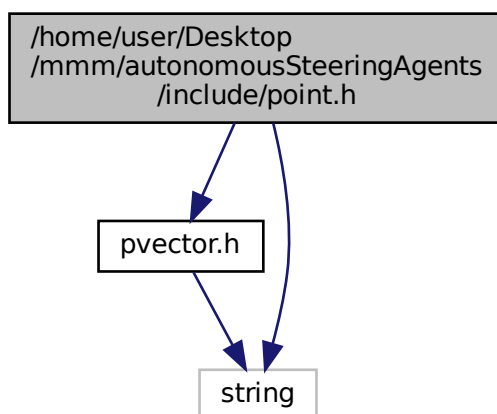
15.05.2021

## 7.23 /home/user/Desktop/mmm/autonomousSteeringAgents/include/point.h File Reference

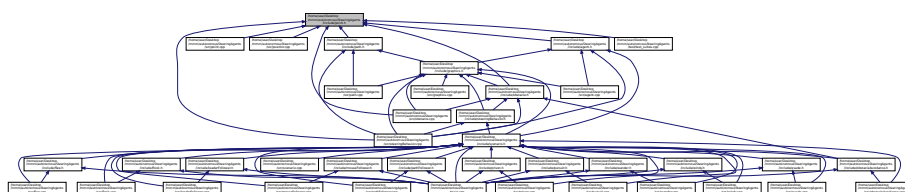
point class used for point operations

```
#include "pvector.h"
#include <string>
```

Include dependency graph for point.h:



This graph shows which files directly or indirectly include this file:



### Classes

- class `point`

### 7.23.1 Detailed Description

point class used for point operations

Author

Mehmet Rıza Öz - [mehmetrizaoz@gmail.com](mailto:mehmetrizaoz@gmail.com)

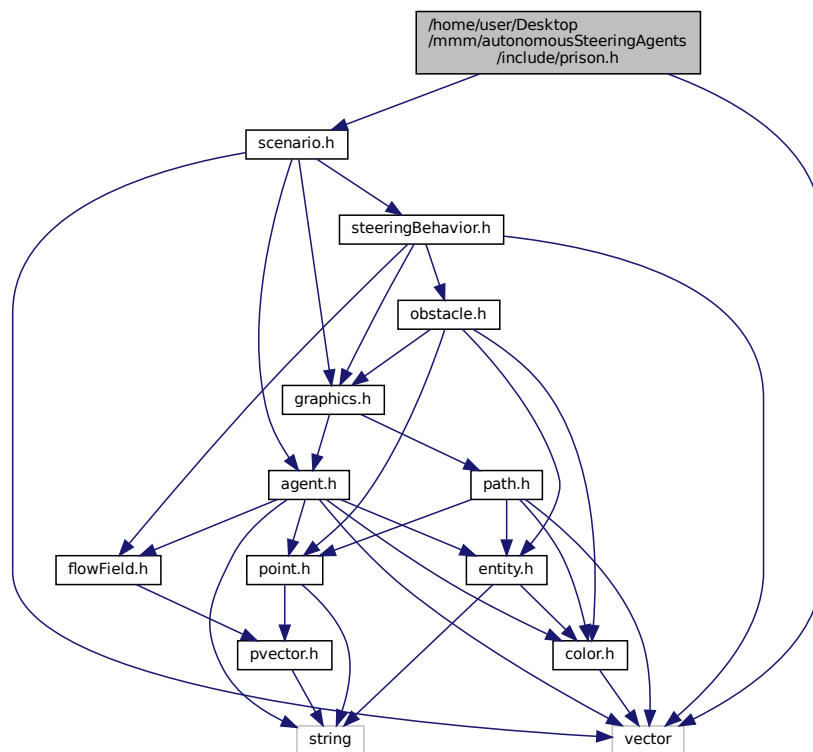
Date

15.05.2021

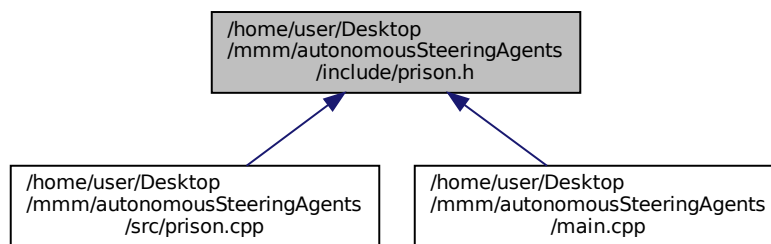
## 7.24 /home/user/Desktop/mmm/autonomousSteeringAgents/include/prison.h File Reference

agents cant escape from field scenario

```
#include "scenario.h"
#include <vector>
Include dependency graph for prison.h:
```



This graph shows which files directly or indirectly include this file:



## Classes

- class [prison](#)

### 7.24.1 Detailed Description

agents cant escape from field scenario

#### Author

Mehmet Rıza Öz - [mehmetrizaoz@gmail.com](mailto:mehmetrizaoz@gmail.com)

#### Date

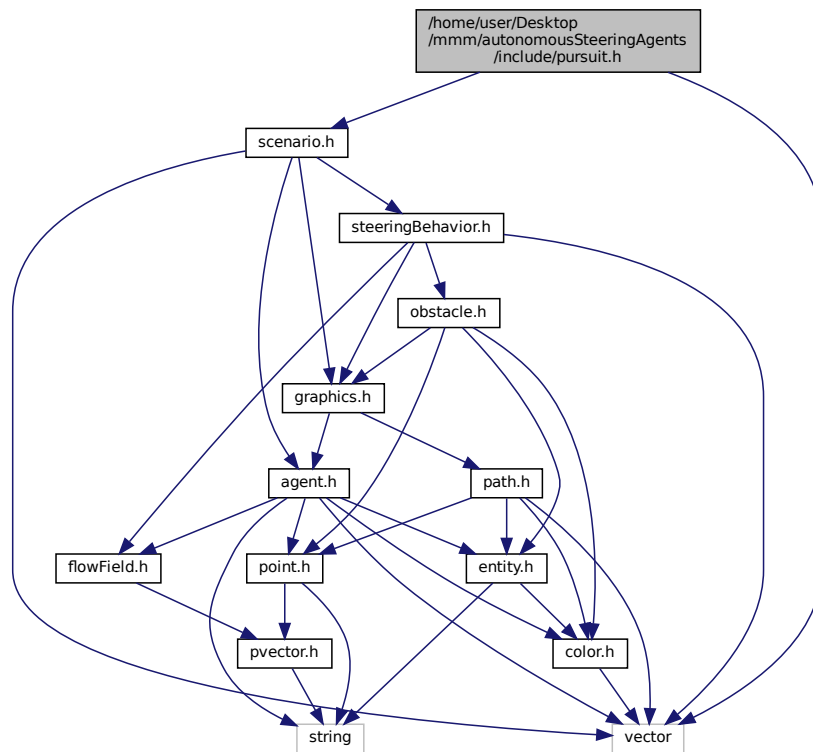
15.05.2021

## 7.25 /home/user/Desktop/mmm/autonomousSteeringAgents/include/pursuit.h File Reference

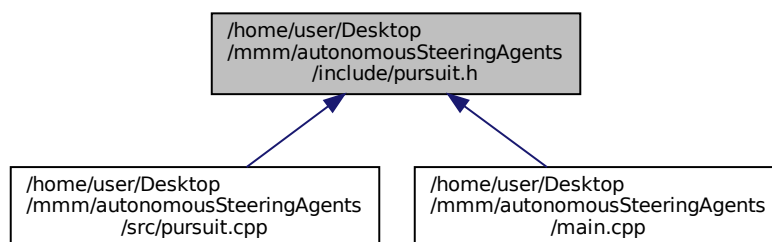
one agent pursue other one scenario

```
#include "scenario.h"
#include <vector>
```

Include dependency graph for `pursuit.h`:



This graph shows which files directly or indirectly include this file:



## Classes

- class [pursuit](#)

### 7.25.1 Detailed Description

one agent pursue other one scenario

## Author

Mehmet Rıza Öz - [mehmetrizaoz@gmail.com](mailto:mehmetrizaoz@gmail.com)

## Date

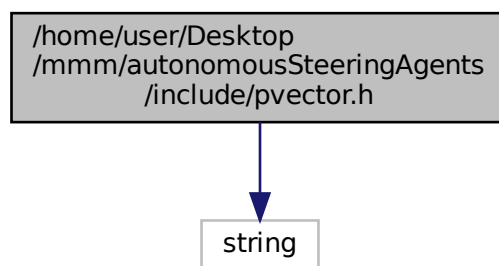
15.05.2021

## 7.26 /home/user/Desktop/mmm/autonomousSteeringAgents/include/pvector.h File Reference

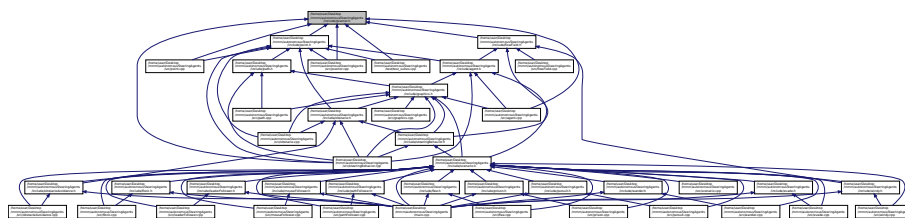
pvector class used for 2D vector operations

```
#include <string>
```

Include dependency graph for pvector.h:



This graph shows which files directly or indirectly include this file:



### Classes

- class `pvector`

### Macros

- `#define PI 3.14159265`

### 7.26.1 Detailed Description

pvector class used for 2D vector operations

#### Author

Mehmet Rıza Öz - [mehmetrizaoz@gmail.com](mailto:mehmetrizaoz@gmail.com)

#### Date

15.05.2021

### 7.26.2 Macro Definition Documentation

#### 7.26.2.1 PI

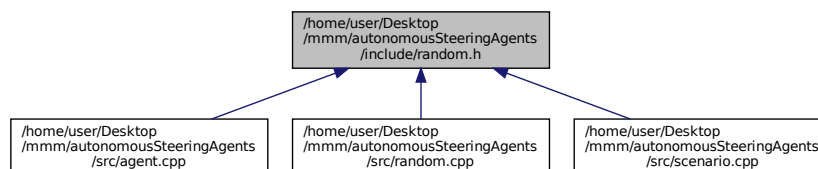
```
#define PI 3.14159265
```

Definition at line 11 of file pvector.h.

## 7.27 [/home/user/Desktop/mmm/autonomousSteeringAgents/include/random.h](#) File Reference

utility class for random operations

This graph shows which files directly or indirectly include this file:



## Classes

- class [random](#)

### 7.27.1 Detailed Description

utility class for random operations

Author

Mehmet Rıza Öz - [mehmetrizaoz@gmail.com](mailto:mehmetrizaoz@gmail.com)

Date

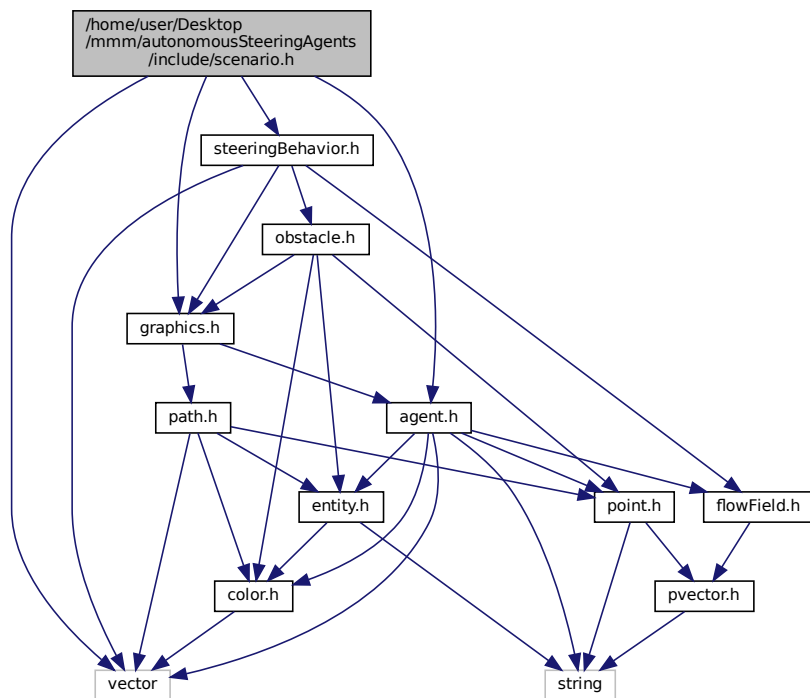
15.05.2021

## 7.28 /home/user/Desktop/mmm/autonomousSteeringAgents/include/scenario.h File Reference

base class for all scenarios

```
#include "agent.h"
#include "graphics.h"
#include "steeringBehavior.h"
#include <vector>
```

Include dependency graph for scenario.h:



This graph shows which files directly or indirectly include this file:



## Classes

- class [scenario](#)

## Enumerations

- enum [types](#) { [RANDOM](#) =0, [STATIC](#), [TROOP](#) }

### 7.28.1 Detailed Description

base class for all scenarios

#### Author

Mehmet Rıza Öz - [mehmetrizaoz@gmail.com](mailto:mehmetrizaoz@gmail.com)

#### Date

15.05.2021

### 7.28.2 Enumeration Type Documentation

#### 7.28.2.1 types

enum [types](#)

#### Enumerator

|        |  |
|--------|--|
| RANDOM |  |
| STATIC |  |
| TROOP  |  |

Definition at line 17 of file scenario.h.

```
17 { RANDOM=0, STATIC, TROOP };
```

## 7.29 /home/user/Desktop/mmm/autonomousSteeringAgents/include/steeringBehavior.h File Reference

functions for autonomous steering behaviors

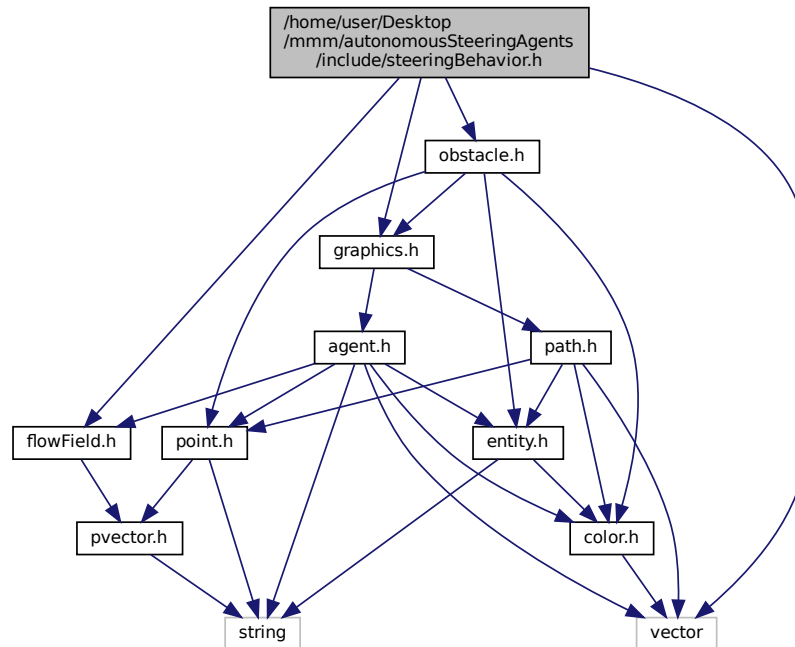
```
#include "flowField.h"
#include <vector>
```



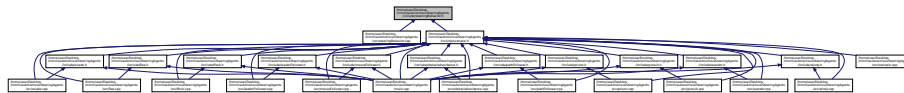
```
#include "graphics.h"
```

```
#include "obstacle.h"
```

Include dependency graph for steeringBehavior.h:



This graph shows which files directly or indirectly include this file:



## Classes

- class [steeringBehavior](#)

## Macros

- `#define CIRCLE_DISTANCE 0.1`
- `#define CIRCLE_RADIUS 0.4`
- `#define FOLLOW_MOUSE 1`
- `#define STAY_IN_FIELD 2`
- `#define IN_FLOW_FIELD 3`
- `#define AVOID_OBSTACLE 4`
- `#define STAY_IN_PATH 5`
- `#define FLOCK 6`
- `#define WANDER 7`
- `#define FLEE 8`
- `#define PURSUIT 9`
- `#define EVADE 10`
- `#define LEADER_FOLLOWER 11`

### 7.29.1 Detailed Description

functions for autonomous steering behaviors

#### Author

Mehmet Rıza Öz - [mehmetrizaoz@gmail.com](mailto:mehmetrizaoz@gmail.com)

#### Date

15.05.2021

### 7.29.2 Macro Definition Documentation

#### 7.29.2.1 AVOID\_OBSTACLE

```
#define AVOID_OBSTACLE 4
```

Definition at line 21 of file steeringBehavior.h.

#### 7.29.2.2 CIRCLE\_DISTANCE

```
#define CIRCLE_DISTANCE 0.1
```

Definition at line 15 of file steeringBehavior.h.

#### 7.29.2.3 CIRCLE\_RADIUS

```
#define CIRCLE_RADIUS 0.4
```

Definition at line 16 of file steeringBehavior.h.

#### 7.29.2.4 EVADE

```
#define EVADE 10
```

Definition at line 27 of file steeringBehavior.h.

#### **7.29.2.5 FLEE**

```
#define FLEE 8
```

Definition at line 25 of file steeringBehavior.h.

#### **7.29.2.6 FLOCK**

```
#define FLOCK 6
```

Definition at line 23 of file steeringBehavior.h.

#### **7.29.2.7 FOLLOW\_MOUSE**

```
#define FOLLOW_MOUSE 1
```

Definition at line 18 of file steeringBehavior.h.

#### **7.29.2.8 IN\_FLOW\_FIELD**

```
#define IN_FLOW_FIELD 3
```

Definition at line 20 of file steeringBehavior.h.

#### **7.29.2.9 LEADER\_FOLLOWER**

```
#define LEADER_FOLLOWER 11
```

Definition at line 28 of file steeringBehavior.h.

#### **7.29.2.10 PURSUIT**

```
#define PURSUIT 9
```

Definition at line 26 of file steeringBehavior.h.

### 7.29.2.11 STAY\_IN\_FIELD

```
#define STAY_IN_FIELD 2
```

Definition at line 19 of file steeringBehavior.h.

### 7.29.2.12 STAY\_IN\_PATH

```
#define STAY_IN_PATH 5
```

Definition at line 22 of file steeringBehavior.h.

### 7.29.2.13 WANDER

```
#define WANDER 7
```

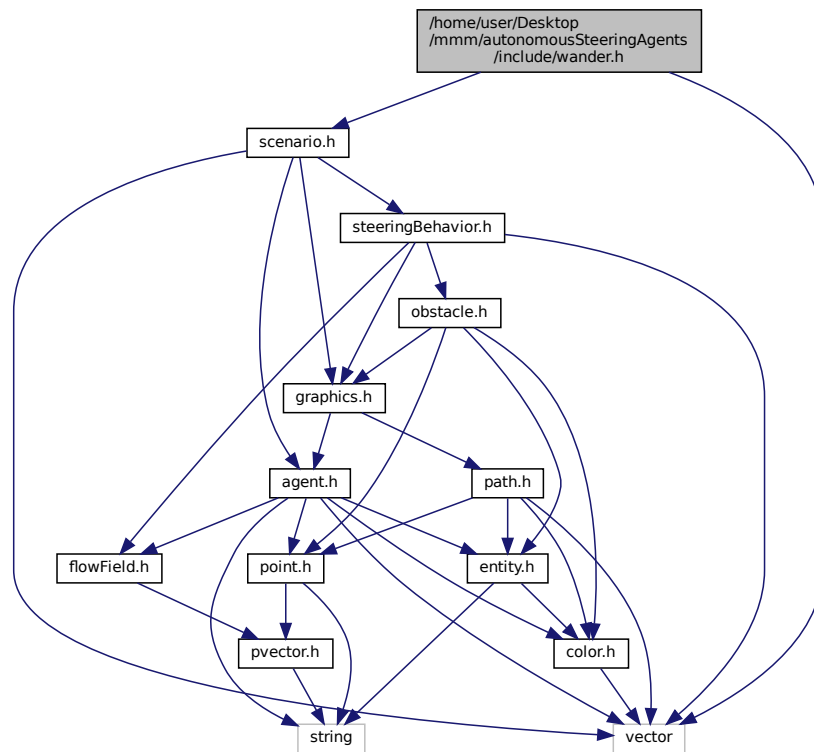
Definition at line 24 of file steeringBehavior.h.

## 7.30 /home/user/Desktop/mmm/autonomousSteeringAgents/include/wander.h File Reference

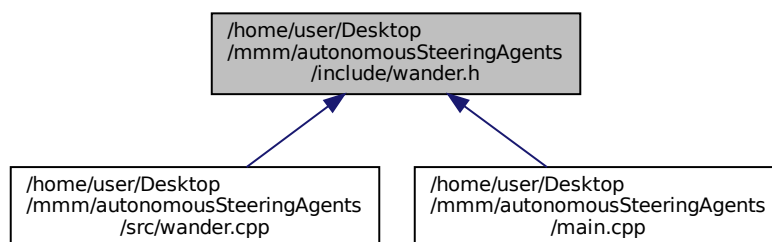
random wandering agents scenario

```
#include "scenario.h"  
#include <vector>
```

Include dependency graph for wander.h:



This graph shows which files directly or indirectly include this file:



## Classes

- class [wander](#)

### 7.30.1 Detailed Description

random wandering agents scenario

## Author

Mehmet Rıza Öz - [mehmetrizaoz@gmail.com](mailto:mehmetrizaoz@gmail.com)

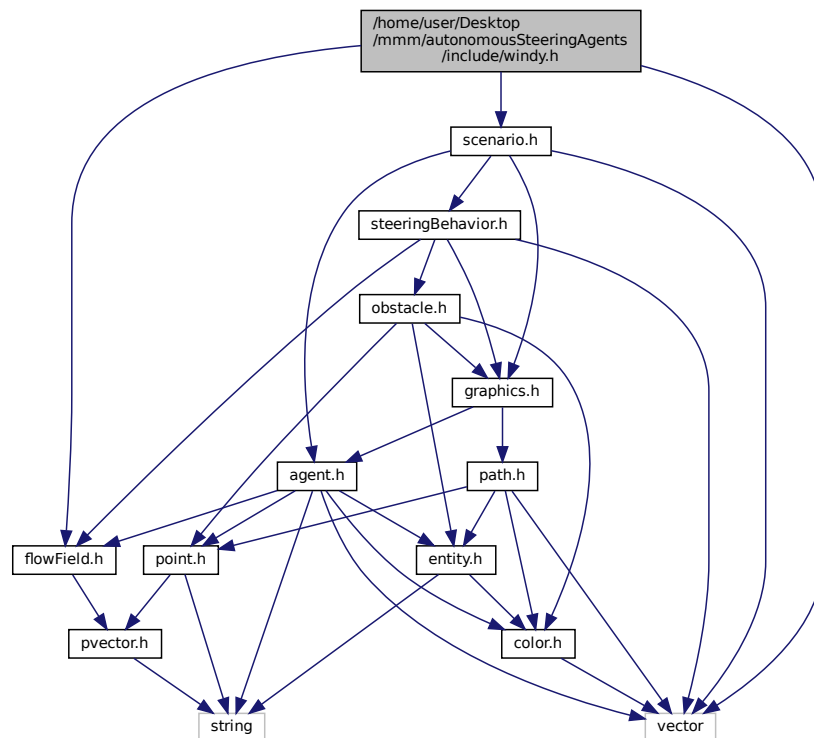
## Date

15.05.2021

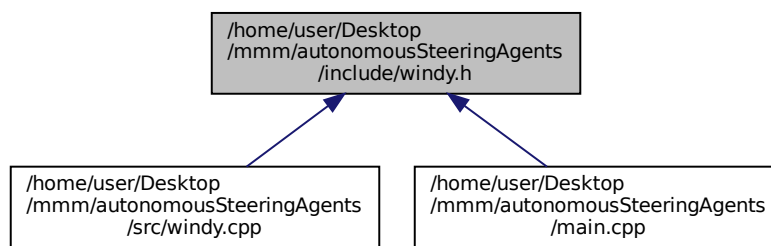
### 7.31 /home/user/Desktop/mmm/autonomousSteeringAgents/include/windy.h File Reference

windy air scenario

```
#include "scenario.h"
#include "flowField.h"
#include <vector>
Include dependency graph for windy.h:
```



This graph shows which files directly or indirectly include this file:



## Classes

- class [windy](#)

### 7.31.1 Detailed Description

windy air scenario

#### Author

Mehmet Rıza Öz - [mehmetrizaoz@gmail.com](mailto:mehmetrizaoz@gmail.com)

#### Date

15.05.2021

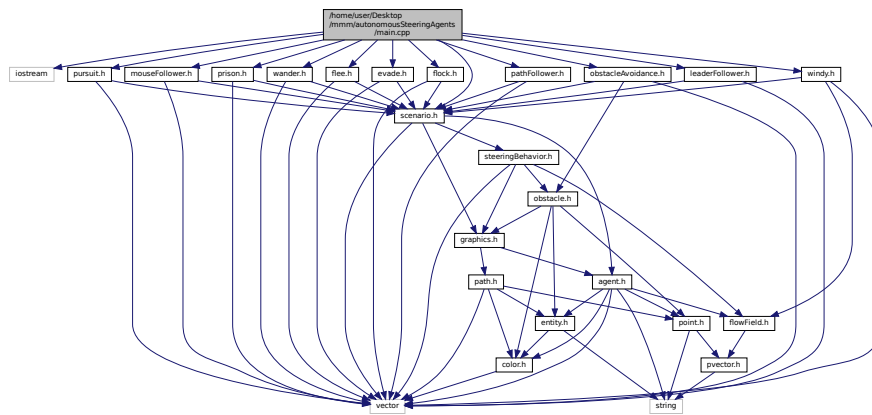
## 7.32 /home/user/Desktop/mmm/autonomousSteeringAgents/leaderFollower.d File Reference

## 7.33 /home/user/Desktop/mmm/autonomousSteeringAgents/main.cpp File Reference

client code

```
#include <iostream>
#include "mouseFollower.h"
#include "prison.h"
#include "windy.h"
#include "wander.h"
#include "pursuit.h"
#include "flee.h"
```

```
#include "scenario.h"
#include "evade.h"
#include "flock.h"
#include "pathFollower.h"
#include "leaderFollower.h"
#include "obstacleAvoidance.h"
Include dependency graph for main.cpp:
```



## Functions

- void `menu` ()  
*displays menu*
- int `main` (int argc, char \*\*argv)  
*main routine*

## Variables

- int `mode`  
*specifies user selected scenario*

### 7.33.1 Detailed Description

client code

Author

Mehmet Rıza Öz - [mehmetrizaoz@gmail.com](mailto:mehmetrizaoz@gmail.com)

Date

15.05.2021

### 7.33.2 Function Documentation



### 7.33.2.1 main()

```
int main (
    int argc,
    char ** argv )
```

main routine

Definition at line 50 of file main.cpp.

```
50 {
51     menu();
52
53     scenario *sc;
54
55     if(mode == FOLLOW_MOUSE){
56         *sc = mouseFollower();
57     }
58     else if(mode == STAY_IN_FIELD){
59         *sc = prison();
60     }
61     else if(mode == IN_FLOW_FIELD){
62         *sc = windy();
63     }
64     else if(mode == WANDER){
65         *sc = wander();
66     }
67     else if(mode == PURSUIT){
68         *sc = pursuit();
69     }
70     else if(mode == FLEE){
71         *sc = flee();
72     }
73     else if(mode == EVADE){
74         *sc = evade();
75     }
76     else if(mode == FLOCK){
77         *sc = flock();
78     }
79     else if(mode == STAY_IN_PATH){
80         *sc = pathFollower();
81     }
82     else if(mode == AVOID_OBSTACLE){
83         *sc = obstacleAvoidance();
84     }
85     else if(mode == LEADER_FOLLOWER){
86         *sc = leaderFollower();
87     }
88
89     sc->initGL(&argc, argv);
90
91     return 0;
92 }
```

### 7.33.2.2 menu()

```
void menu ( )
```

displays menu

Definition at line 32 of file main.cpp.

```
32 {
33     cout << "Follow Mouse      : 1" << endl;
34     cout << "Stay in Field      : 2" << endl;
35     cout << "In Flow Field      : 3" << endl;
36     cout << "Avoid Obstacles    : 4" << endl;
37     cout << "Stay in Path       : 5" << endl;
38     cout << "FLOCK              : 6" << endl;
39     cout << "WANDER             : 7" << endl;
40     cout << "FLEE               : 8" << endl;
41     cout << "PURSUIT            : 9" << endl;
42     cout << "EVADE              : 10" << endl;
43     cout << "Follow Leader      : 11" << endl;
44     cin >> mode;
45 }
```

### 7.33.3 Variable Documentation

#### 7.33.3.1 mode

```
int mode
```

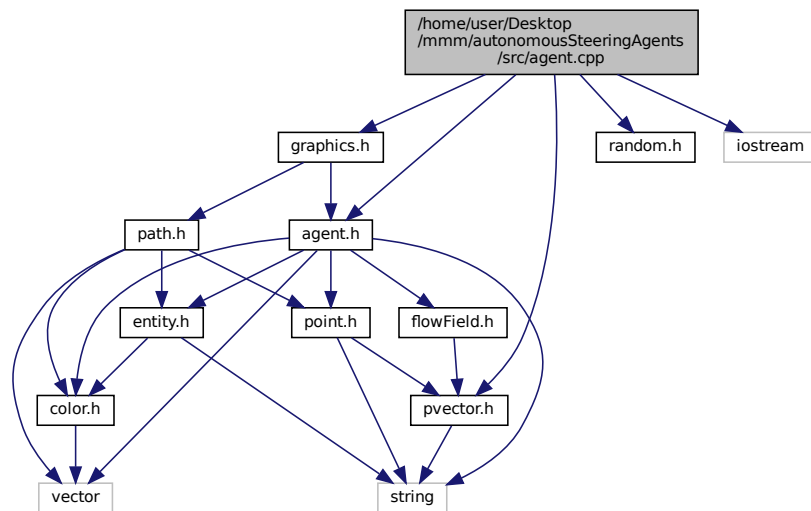
specifies user selected scenario

Definition at line 27 of file main.cpp.



- 7.34** `/home/user/Desktop/mmm/autonomousSteeringAgents/main.d` File Reference
- 7.35** `/home/user/Desktop/mmm/autonomousSteeringAgents/mouse↵Follower.d` File Reference
- 7.36** `/home/user/Desktop/mmm/autonomousSteeringAgents/obstacle.d` File Reference
- 7.37** `/home/user/Desktop/mmm/autonomousSteeringAgents/obstacle↵Avoidance.d` File Reference
- 7.38** `/home/user/Desktop/mmm/autonomousSteeringAgents/path.d` File Reference
- 7.39** `/home/user/Desktop/mmm/autonomousSteeringAgents/path↵Follower.d` File Reference
- 7.40** `/home/user/Desktop/mmm/autonomousSteeringAgents/point.d` File Reference
- 7.41** `/home/user/Desktop/mmm/autonomousSteeringAgents/prison.d` File Reference
- 7.42** `/home/user/Desktop/mmm/autonomousSteeringAgents/pursuit.d` File Reference
- 7.43** `/home/user/Desktop/mmm/autonomousSteeringAgents/pvector.d` File Reference
- 7.44** `/home/user/Desktop/mmm/autonomousSteeringAgents/random.d` File Reference
- 7.45** `/home/user/Desktop/mmm/autonomousSteeringAgents/READ↵ME.md` File Reference
- 7.46** `/home/user/Desktop/mmm/autonomousSteeringAgents/scenario.d` File Reference
- 7.47** `/home/user/Desktop/mmm/autonomousSteering↵Agents/src/agent.cpp` File Reference

```
#include "agent.h"
#include "pvector.h"
#include "graphics.h"
#include "random.h"
#include <iostream>
Include dependency graph for agent.cpp:
```



### 7.47.1 Detailed Description

implementation of the agent class

#### Author

Mehmet Rıza Öz - [mehmetrizaoz@gmail.com](mailto:mehmetrizaoz@gmail.com)

#### Date

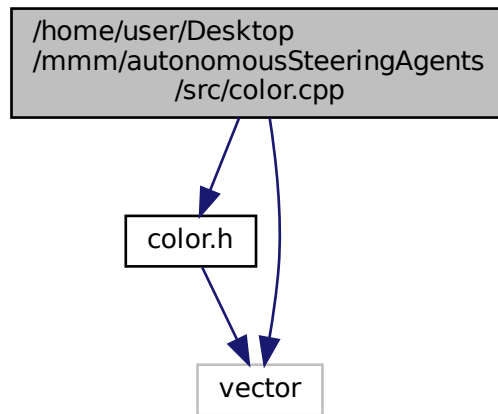
14.05.2021

## 7.48 /home/user/Desktop/mmm/autonomousSteeringAgents/src/color.cpp File Reference

color class implementation

```
#include "color.h"
#include <vector>
```

Include dependency graph for color.cpp:



### 7.48.1 Detailed Description

color class implementation

#### Author

Mehmet Rıza Öz - [mehmetrizaoz@gmail.com](mailto:mehmetrizaoz@gmail.com)

#### Date

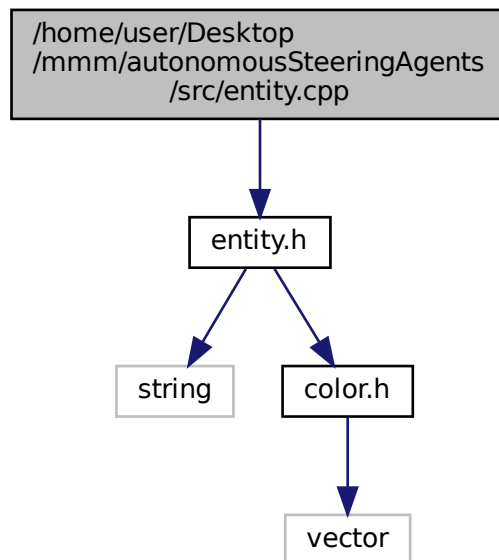
13.05.2021

## 7.49 `/home/user/Desktop/mmm/autonomousSteeringAgents/src/entity.cpp` File Reference

entity class implementation

```
#include "entity.h"
```

Include dependency graph for entity.cpp:



### 7.49.1 Detailed Description

entity class implementation

#### Author

Mehmet Rıza Öz - [mehmetrizaoz@gmail.com](mailto:mehmetrizaoz@gmail.com)

#### Date

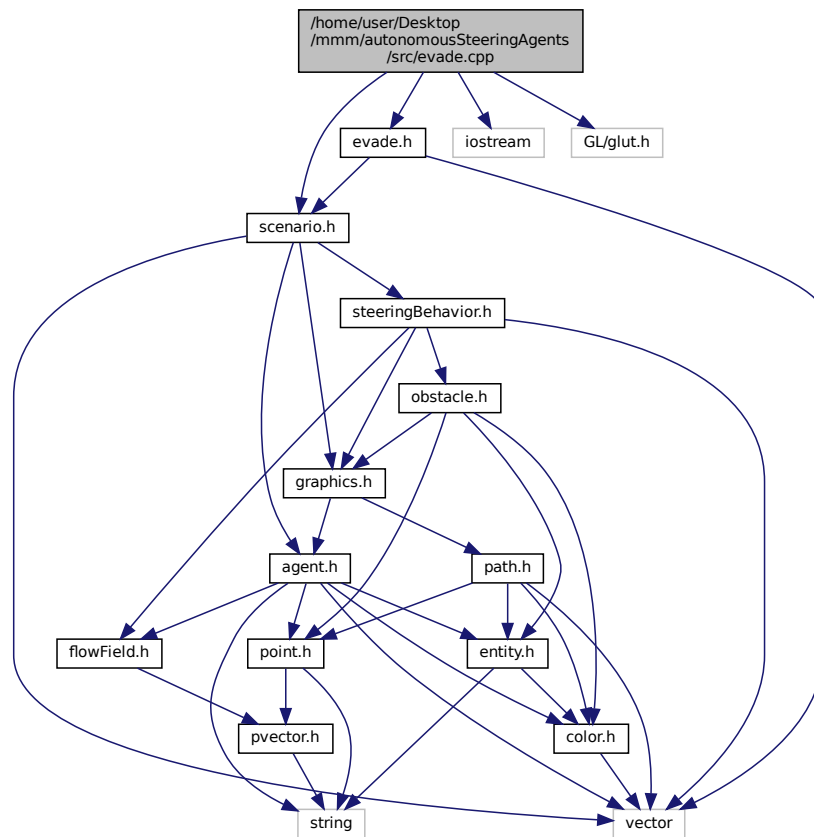
18.05.2021

## 7.50 /home/user/Desktop/mmm/autonomousSteeringAgents/src/evade.cpp File Reference

evade class implementation

```
#include "scenario.h"
#include "evade.h"
#include <iostream>
```

```
#include <GL/glut.h>
Include dependency graph for evade.cpp:
```



### 7.50.1 Detailed Description

evade class implementation

Author

Mehmet Rıza Öz - [mehmetrizaoz@gmail.com](mailto:mehmetrizaoz@gmail.com)

Date

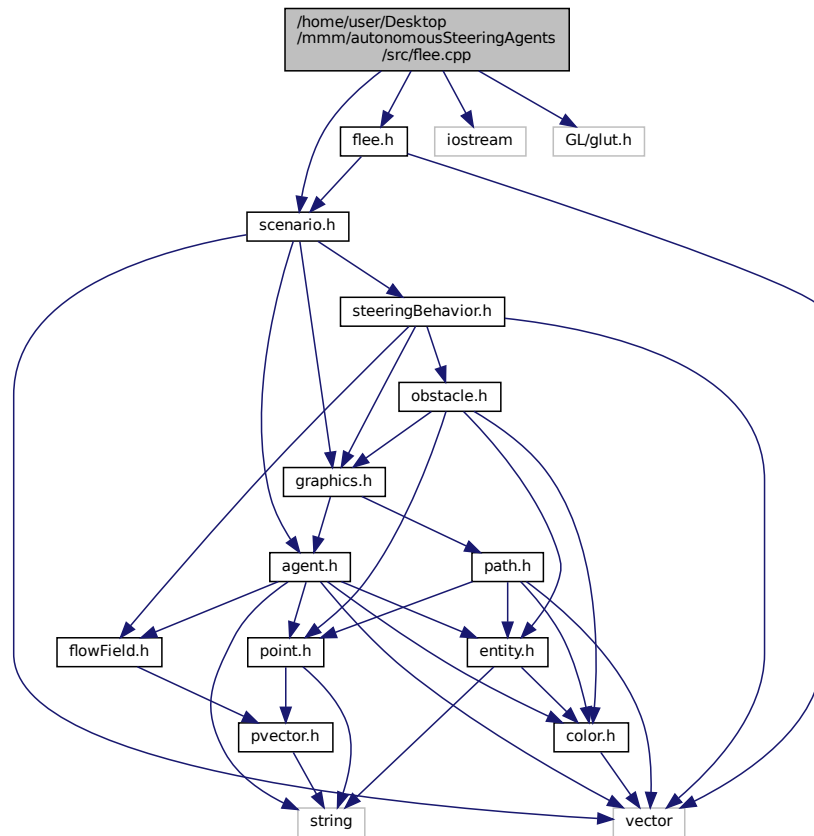
15.05.2021

## 7.51 /home/user/Desktop/mmm/autonomousSteeringAgents/src/flee.cpp File Reference

flee class implementation



```
#include "scenario.h"
#include "flee.h"
#include <iostream>
#include <GL/glut.h>
Include dependency graph for flee.cpp:
```



### 7.51.1 Detailed Description

flee class implementation

Author

Mehmet Rıza Öz - [mehmetrizaoz@gmail.com](mailto:mehmetrizaoz@gmail.com)

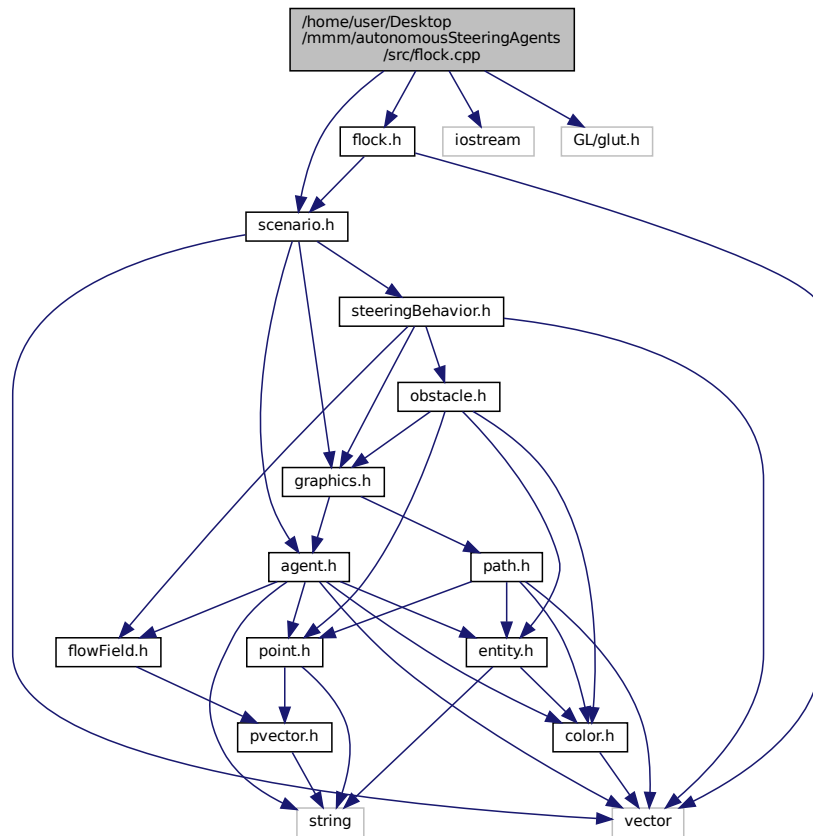
Date

15.05.2021

## 7.52 /home/user/Desktop/mmm/autonomousSteeringAgents/src/flock.cpp File Reference

flock class implementation

```
#include "scenario.h"
#include "flock.h"
#include <iostream>
#include <GL/glut.h>
Include dependency graph for flock.cpp:
```



### 7.52.1 Detailed Description

flock class implementation

Author

Mehmet Rıza Öz - [mehmetrizaoz@gmail.com](mailto:mehmetrizaoz@gmail.com)

Date

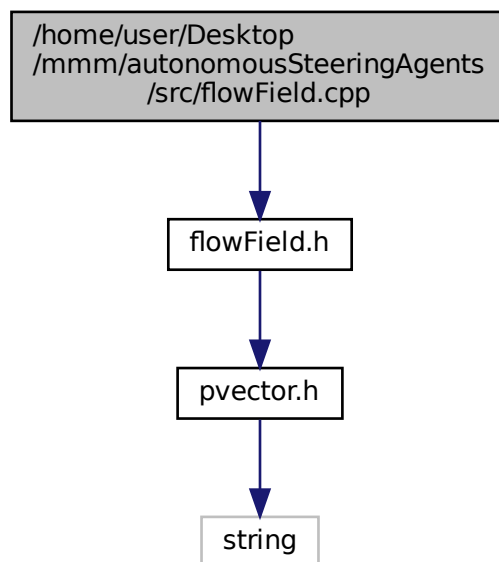
15.05.2021

## 7.53 /home/user/Desktop/mmm/autonomousSteeringAgents/src/flowField.cpp File Reference

[flowField](#) class implementation

```
#include "flowField.h"
```

Include dependency graph for flowField.cpp:



### 7.53.1 Detailed Description

`flowField` class implementation

#### Author

Mehmet Rıza Öz - [mehmetrizaoz@gmail.com](mailto:mehmetrizaoz@gmail.com)

#### Date

13.05.2021

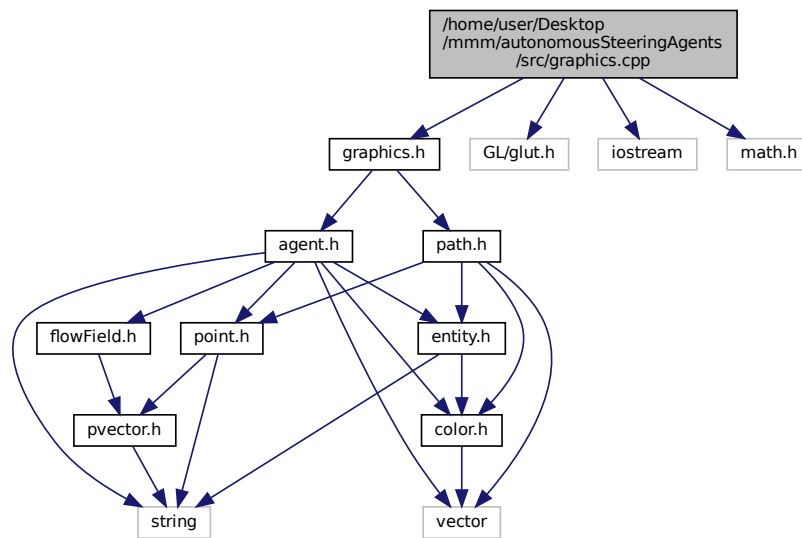
## 7.54 /home/user/Desktop/mmm/autonomousSteeringAgents/src/graphics.cpp File Reference

`graphics` class implementation

```
#include "graphics.h"
#include <GL/glut.h>
#include <iostream>
```

```
#include "math.h"
```

Include dependency graph for graphics.cpp:



### 7.54.1 Detailed Description

graphics class implementation

Author

Mehmet Rıza Öz - [mehmetrizaoz@gmail.com](mailto:mehmetrizaoz@gmail.com)

Date

15.05.2021

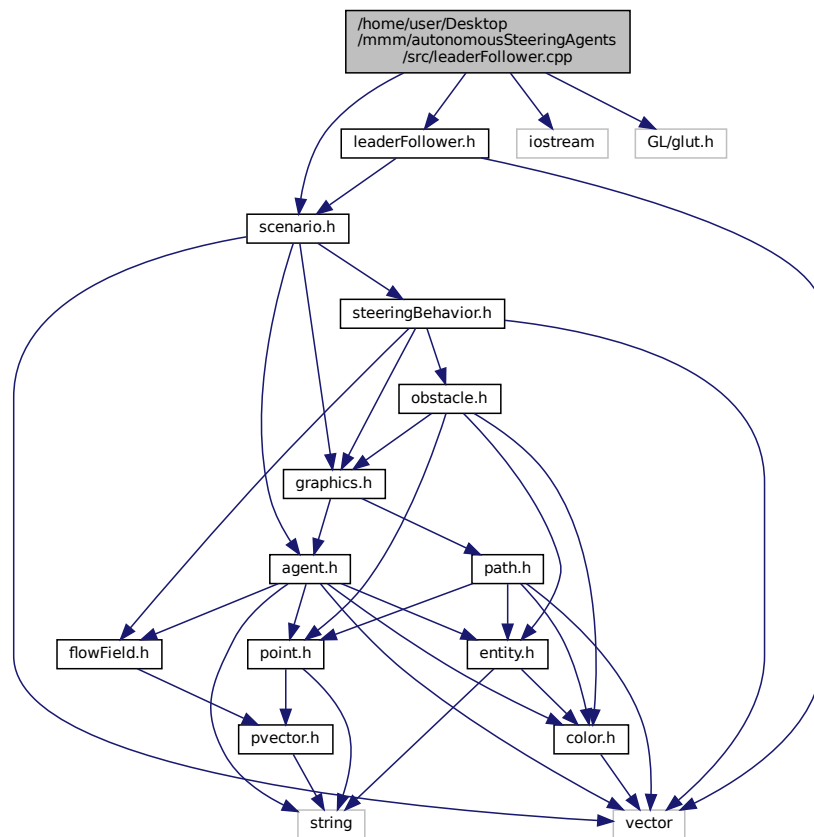
## 7.55 /home/user/Desktop/mmm/autonomousSteeringAgents/src/leaderFollower.cpp File Reference

[leaderFollower](#) class implementation

```
#include "scenario.h"
#include "leaderFollower.h"
#include <iostream>
```

```
#include <GL/glut.h>
```

Include dependency graph for leaderFollower.cpp:



### 7.55.1 Detailed Description

[leaderFollower](#) class implementation

Author

Mehmet Rıza Öz - [mehmetrizaoz@gmail.com](mailto:mehmetrizaoz@gmail.com)

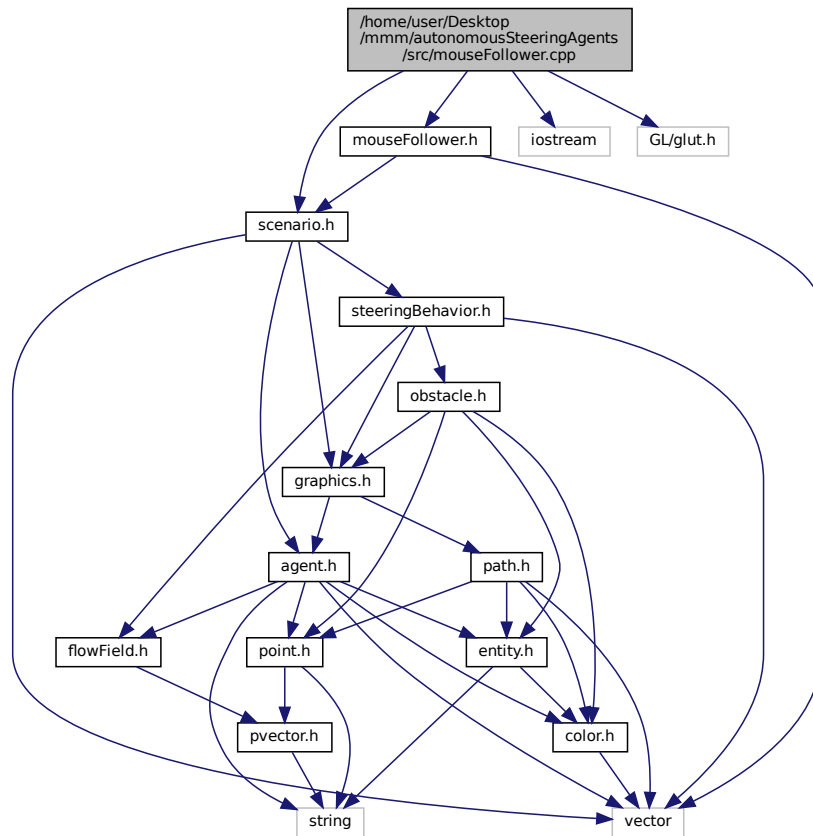
Date

19.05.2021

## 7.56 /home/user/Desktop/mmm/autonomousSteeringAgents/src/mouseFollower.cpp File Reference

[mouseFollower](#) class implementation

```
#include "scenario.h"
#include "mouseFollower.h"
#include <iostream>
#include <GL/glut.h>
Include dependency graph for mouseFollower.cpp:
```



### 7.56.1 Detailed Description

[mouseFollower](#) class implementation

Author

Mehmet Rıza Öz - [mehmetrizaoz@gmail.com](mailto:mehmetrizaoz@gmail.com)

Date

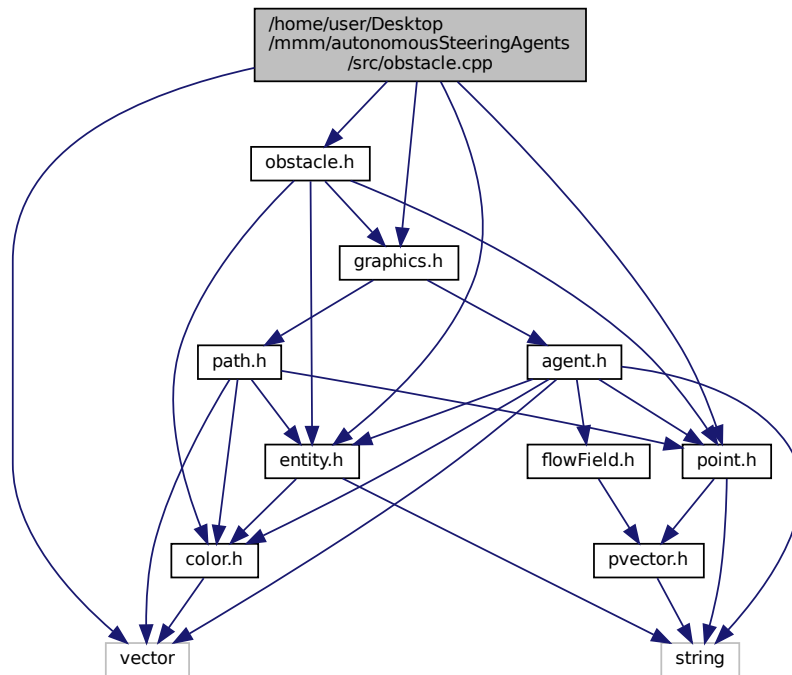
15.05.2021

## 7.57 /home/user/Desktop/mmm/autonomousSteeringAgents/src/obstacle.cpp File Reference

obstacle class implementation

```
#include "obstacle.h"
#include "graphics.h"
#include "point.h"
#include "entity.h"
#include <vector>
```

Include dependency graph for obstacle.cpp:



### 7.57.1 Detailed Description

obstacle class implementation

Author

Mehmet Rıza Öz - [mehmetrizaoz@gmail.com](mailto:mehmetrizaoz@gmail.com)

Date

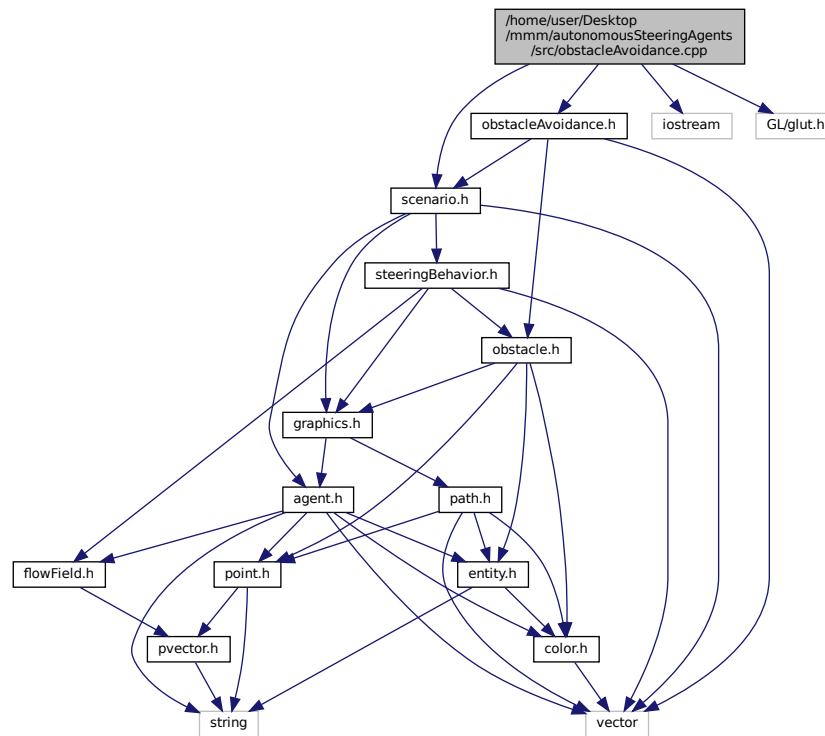
12.05.2021

## 7.58 /home/user/Desktop/mmm/autonomousSteeringAgents/src/obstacleAvoidance.cpp File Reference

[obstacleAvoidance](#) class implementation

```
#include "scenario.h"
#include "obstacleAvoidance.h"
#include <iostream>
#include <GL/glut.h>
```

Include dependency graph for obstacleAvoidance.cpp:



### 7.58.1 Detailed Description

[obstacleAvoidance](#) class implementation

Author

Mehmet Rıza Öz - [mehmetrizaoz@gmail.com](mailto:mehmetrizaoz@gmail.com)

Date

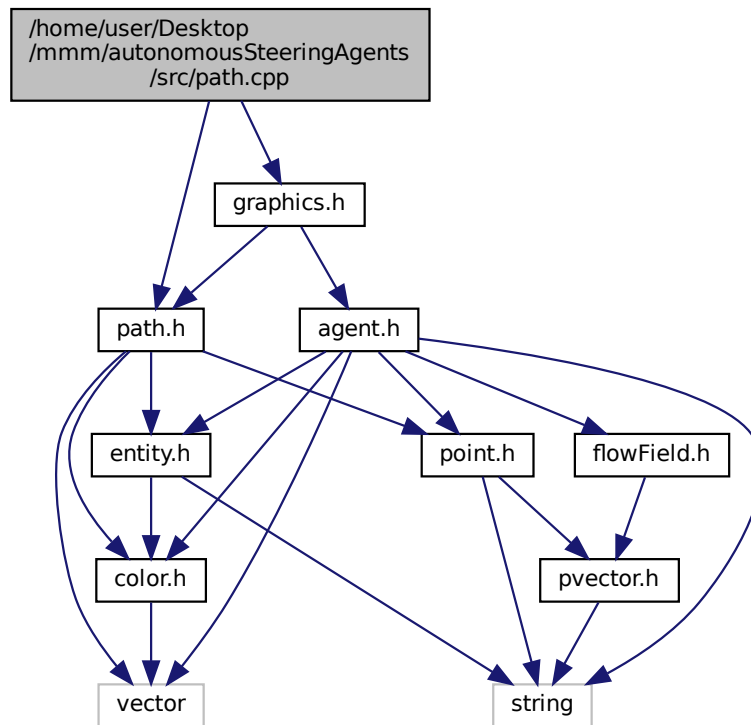
15.05.2021

## 7.59 /home/user/Desktop/mmm/autonomousSteeringAgents/src/path.cpp File Reference

[path](#) class implementation



```
#include "path.h"
#include "graphics.h"
Include dependency graph for path.cpp:
```



### 7.59.1 Detailed Description

path class implementation

Author

Mehmet Rıza Öz - [mehmetrizaoz@gmail.com](mailto:mehmetrizaoz@gmail.com)

Date

12.05.2021

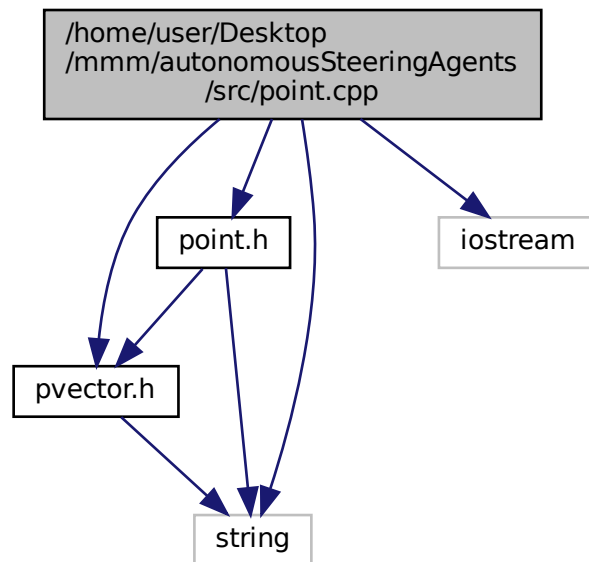
## 7.60 /home/user/Desktop/mmm/autonomousSteeringAgents/src/pathFollower.cpp File Reference

[pathFollower](#) class implementation



```
#include "point.h"
#include "pvector.h"
#include <string>
#include <iostream>
```

Include dependency graph for point.cpp:



### 7.61.1 Detailed Description

point class implementation file

#### Author

Mehmet Rıza Öz - [mehmetrizaoz@gmail.com](mailto:mehmetrizaoz@gmail.com)

#### Date

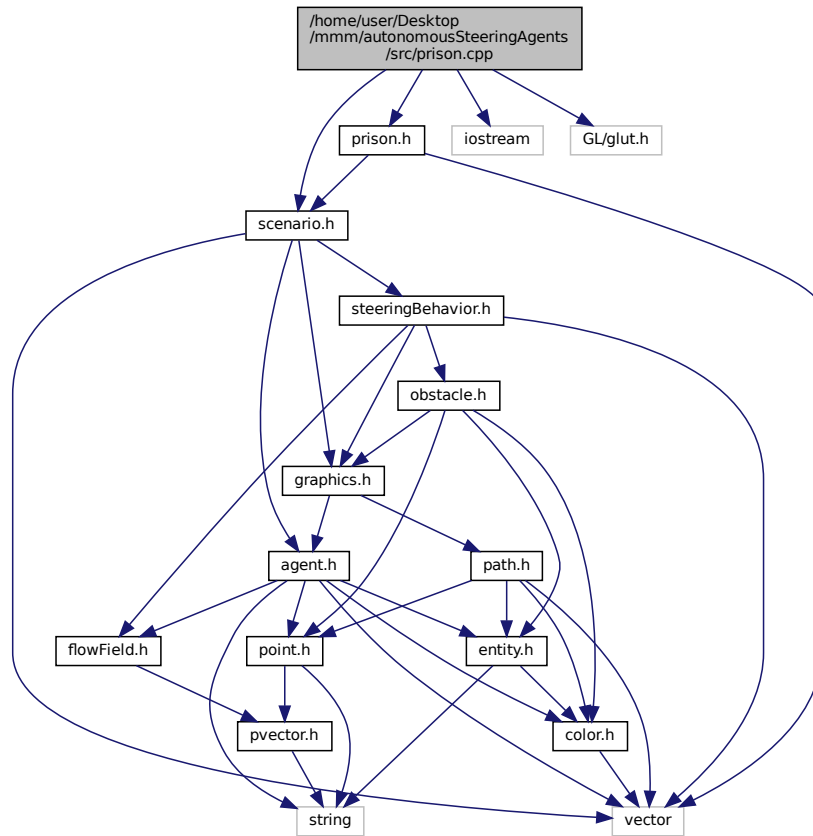
15.05.2021

## 7.62 /home/user/Desktop/mmm/autonomousSteeringAgents/src/prison.cpp File Reference

prison class implementation

```
#include "scenario.h"
#include "prison.h"
```

```
#include <iostream>
#include <GL/glut.h>
Include dependency graph for prison.cpp:
```



## Macros

- #define WALL 30
- #define DISTANCE 2

### 7.62.1 Detailed Description

prison class implementation

Author

Mehmet Rıza Öz - [mehmetrizaoz@gmail.com](mailto:mehmetrizaoz@gmail.com)

Date

15.05.2021

## 7.62.2 Macro Definition Documentation

### 7.62.2.1 DISTANCE

```
#define DISTANCE 2
```

Definition at line 14 of file prison.cpp.

### 7.62.2.2 WALL

```
#define WALL 30
```

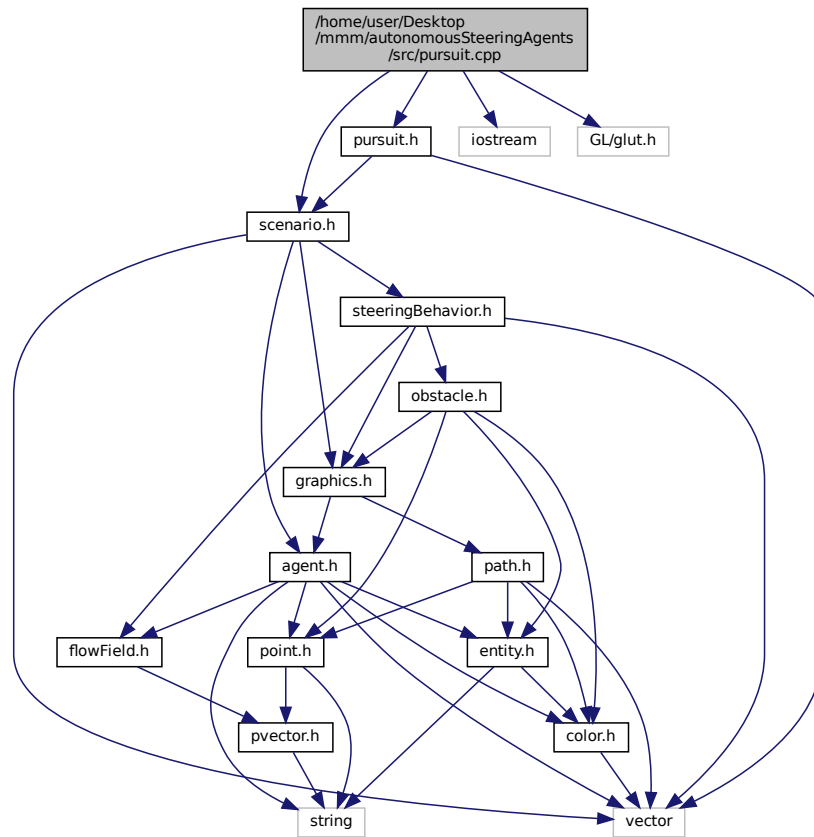
Definition at line 13 of file prison.cpp.

## 7.63 /home/user/Desktop/mmm/autonomousSteeringAgents/src/pursuit.cpp File Reference

prison class implementation

```
#include "scenario.h"
#include "pursuit.h"
#include <iostream>
#include <GL/glut.h>
```

Include dependency graph for `pursuit.cpp`:



### 7.63.1 Detailed Description

prison class implementation

Author

Mehmet Rıza Öz - [mehmetrizaoz@gmail.com](mailto:mehmetrizaoz@gmail.com)

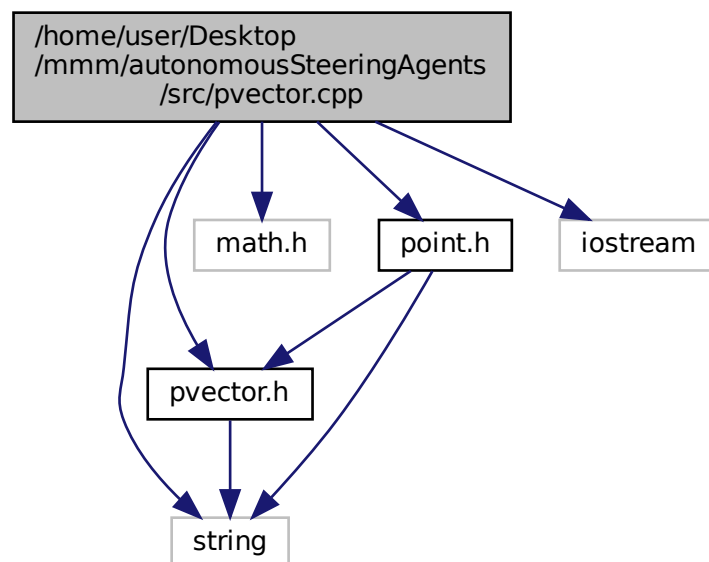
Date

15.05.2021

## 7.64 /home/user/Desktop/mmm/autonomousSteeringAgents/src/pvector.cpp File Reference

pvector class implementation

```
#include "pvector.h"
#include "math.h"
#include "point.h"
#include <iostream>
#include <string>
Include dependency graph for pvector.cpp:
```



### 7.64.1 Detailed Description

pvector class implementation

#### Author

Mehmet Rıza Öz - [mehmetrizaoz@gmail.com](mailto:mehmetrizaoz@gmail.com)

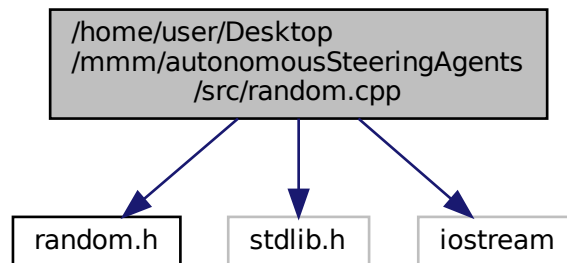
#### Date

15.05.2021

## 7.65 /home/user/Desktop/mmm/autonomousSteeringAgents/src/random.cpp File Reference

utility class for random operations

```
#include "random.h"
#include <stdlib.h>
#include <iostream>
Include dependency graph for random.cpp:
```



### 7.65.1 Detailed Description

utility class for random operations

#### Author

Mehmet Rıza Öz - [mehmetrizaoz@gmail.com](mailto:mehmetrizaoz@gmail.com)

#### Date

15.05.2021

## 7.66 `/home/user/Desktop/mmm/autonomousSteeringAgents/src/scenario.cpp` File Reference

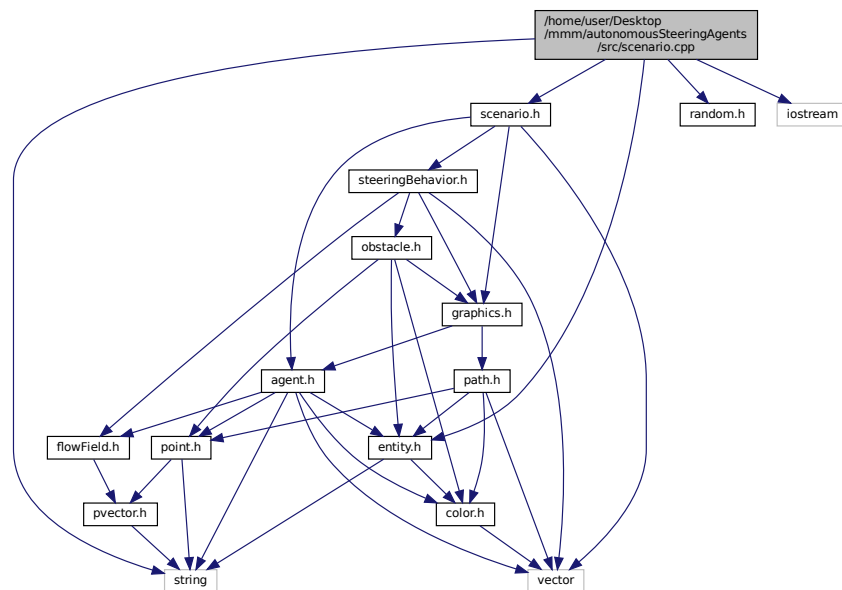
scenario base class implementation

```
#include "scenario.h"
#include "random.h"
#include "entity.h"
#include <iostream>
```



```
#include <string>
```

Include dependency graph for scenario.cpp:



## Macros

- `#define MAX_NUMBER_OF_AGENTS 50`

### 7.66.1 Detailed Description

scenario base class implementation

#### Author

Mehmet Rıza Öz - [mehmetrizaoz@gmail.com](mailto:mehmetrizaoz@gmail.com)

#### Date

15.05.2021

### 7.66.2 Macro Definition Documentation

#### 7.66.2.1 MAX\_NUMBER\_OF\_AGENTS

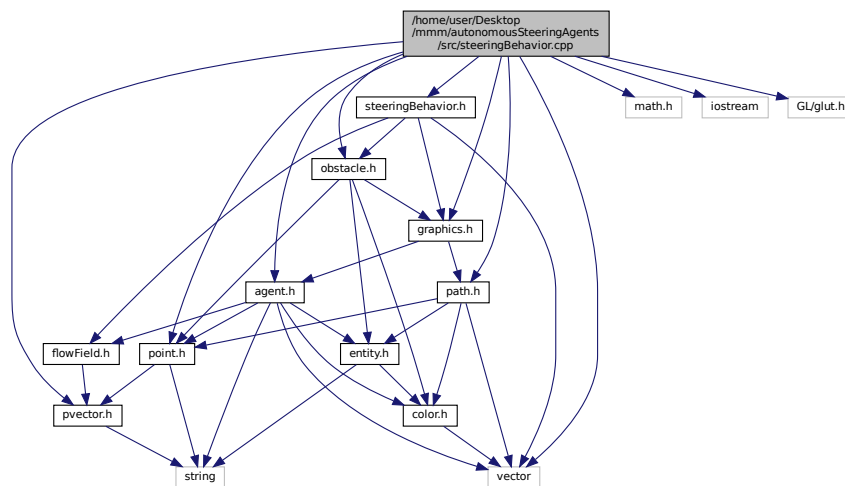
```
#define MAX_NUMBER_OF_AGENTS 50
```

Definition at line 14 of file scenario.cpp.

## 7.67 /home/user/Desktop/mmm/autonomousSteeringAgents/src/steeringBehavior.cpp File Reference

implementation of autonomous steering behaviors

```
#include "steeringBehavior.h"
#include "pvector.h"
#include "agent.h"
#include "path.h"
#include "point.h"
#include <vector>
#include "graphics.h"
#include "math.h"
#include "obstacle.h"
#include <iostream>
#include <GL/glut.h>
Include dependency graph for steeringBehavior.cpp:
```



### 7.67.1 Detailed Description

implementation of autonomous steering behaviors

Author

Mehmet Rıza Öz - [mehmetrizaoz@gmail.com](mailto:mehmetrizaoz@gmail.com)

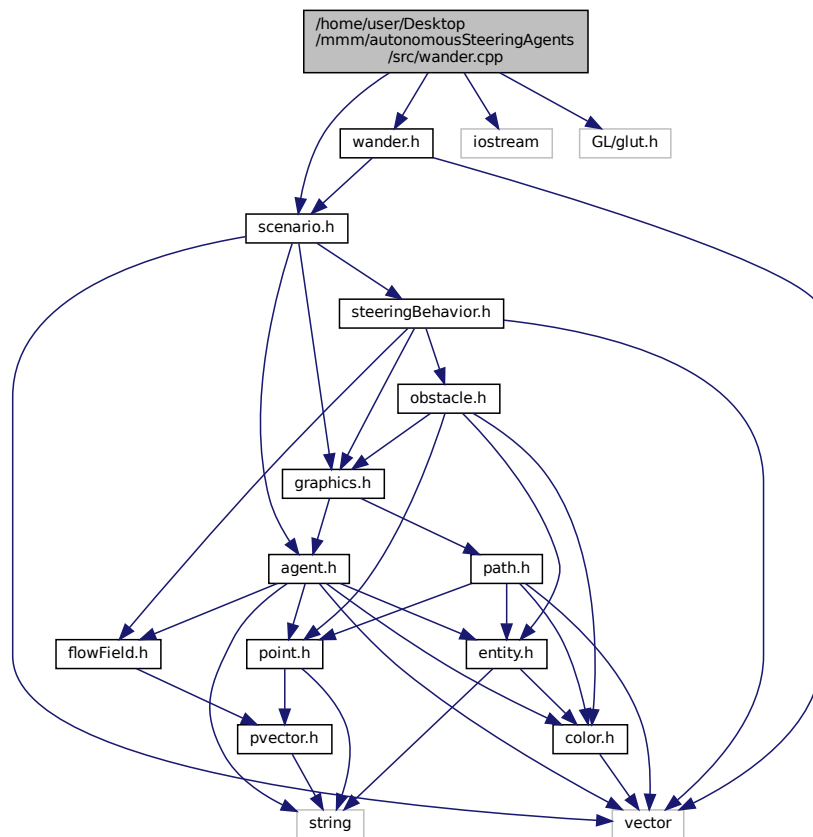
Date

15.05.2021

## 7.68 /home/user/Desktop/mmm/autonomousSteeringAgents/src/wander.cpp File Reference

wander class implementation

```
#include "scenario.h"
#include "wander.h"
#include <iostream>
#include <GL/glut.h>
Include dependency graph for wander.cpp:
```



### 7.68.1 Detailed Description

wander class implementation

Author

Mehmet Rıza Öz - [mehmetrizaoz@gmail.com](mailto:mehmetrizaoz@gmail.com)

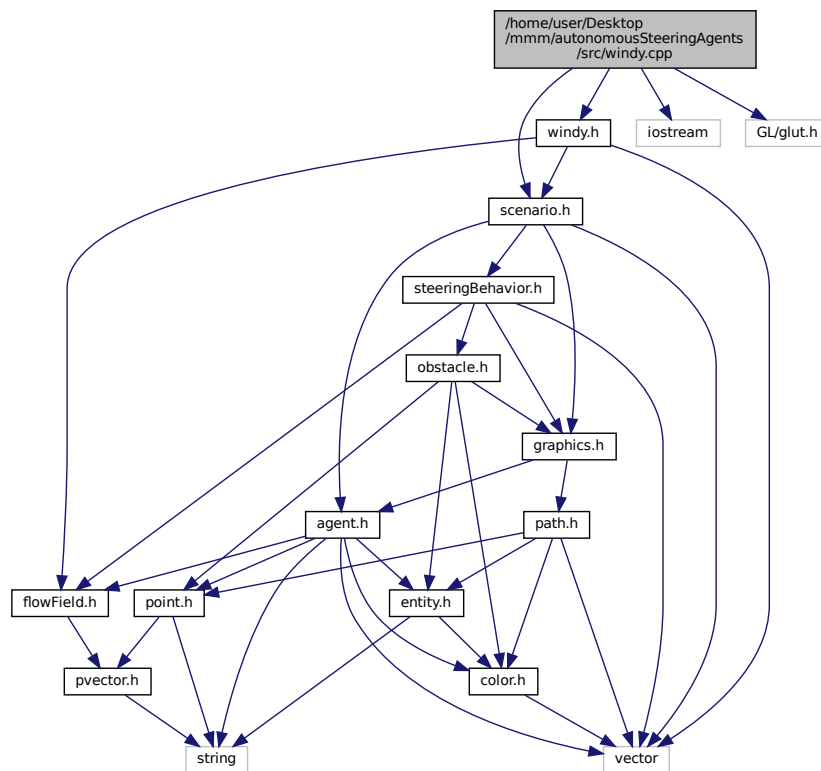
Date

15.05.2021

## 7.69 /home/user/Desktop/mmm/autonomousSteeringAgents/src/windy.cpp File Reference

windy class implementation

```
#include "scenario.h"
#include "windy.h"
#include <iostream>
#include <GL/glut.h>
Include dependency graph for windy.cpp:
```



### 7.69.1 Detailed Description

windy class implementation

Author

Mehmet Rıza Öz - [mehmetrizaoz@gmail.com](mailto:mehmetrizaoz@gmail.com)

Date

15.05.2021

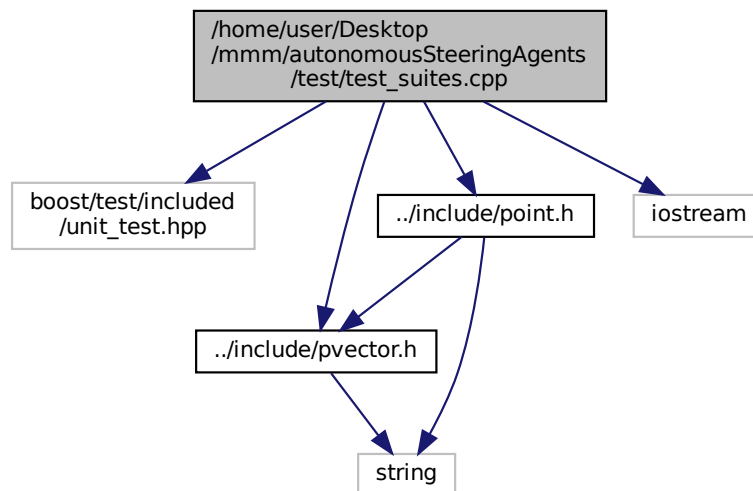
## 7.70 /home/user/Desktop/mmm/autonomousSteeringAgents/steeringBehavior.d File Reference

## 7.71 /home/user/Desktop/mmm/autonomousSteeringAgents/test/test\_suites.cpp File Reference

unit test suites

```
#include <boost/test/included/unit_test.hpp>
#include "../include/pvector.h"
#include "../include/point.h"
#include <iostream>
```

Include dependency graph for test\_suites.cpp:



### Macros

- `#define BOOST_TEST_MODULE test_suites`

### Functions

- `BOOST_AUTO_TEST_CASE (s1t1)`  
*pvector magnitude test case*
- `BOOST_AUTO_TEST_CASE (s1t2)`  
*pvector mul test case*
- `BOOST_AUTO_TEST_CASE (s1t3)`  
*pvector div test case*

- [BOOST\\_AUTO\\_TEST\\_CASE](#) (s1t4)  
*pvector dotproduct test case*
- [BOOST\\_AUTO\\_TEST\\_CASE](#) (s1t5)  
*pvector angle between vectors test case*
- [BOOST\\_AUTO\\_TEST\\_CASE](#) (s1t6)  
*pvector get vector angle test case*
- [BOOST\\_AUTO\\_TEST\\_CASE](#) (s1t7)  
*pvector normalize test case*
- [BOOST\\_AUTO\\_TEST\\_CASE](#) (s1t8)  
*pvector limit test case*
- [BOOST\\_AUTO\\_TEST\\_CASE](#) (s1t9)  
*pvector overloaded operators test case*
- [BOOST\\_AUTO\\_TEST\\_CASE](#) (s2t1)  
*point multiplication test case*
- [BOOST\\_AUTO\\_TEST\\_CASE](#) (s2t2)  
*point division test case*
- [BOOST\\_AUTO\\_TEST\\_CASE](#) (s2t3)  
*point overloaded operators test case*

### 7.71.1 Detailed Description

unit test suites

Author

Mehmet Rıza Öz - [mehmetrizaoz@gmail.com](mailto:mehmetrizaoz@gmail.com)

Date

15.05.2021

### 7.71.2 Macro Definition Documentation

#### 7.71.2.1 BOOST\_TEST\_MODULE

```
#define BOOST_TEST_MODULE test_suites
```

Definition at line 8 of file test\_suites.cpp.

### 7.71.3 Function Documentation

### 7.71.3.1 BOOST\_AUTO\_TEST\_CASE() [1/12]

```
BOOST_AUTO_TEST_CASE (
    slt1 )
```

pvector magnitude test case

Definition at line 22 of file test\_suites.cpp.

```
23 {
24     pvector p1 = pvector(0, 4);
25     pvector p2 = pvector(3, 0);
26     pvector p3 = p1 + p2;
27     BOOST_CHECK(p3.magnitude() == 5);
28 }
```

### 7.71.3.2 BOOST\_AUTO\_TEST\_CASE() [2/12]

```
BOOST_AUTO_TEST_CASE (
    slt2 )
```

pvector mul test case

Definition at line 33 of file test\_suites.cpp.

```
34 {
35     pvector p1 = pvector(1, 1);
36     p1.mul(3);
37     pvector p2 = pvector(3, 3);
38     BOOST_CHECK(p1 == p2);
39 }
```

### 7.71.3.3 BOOST\_AUTO\_TEST\_CASE() [3/12]

```
BOOST_AUTO_TEST_CASE (
    slt3 )
```

pvector div test case

Definition at line 44 of file test\_suites.cpp.

```
45 {
46     pvector p1 = pvector(5, 5);
47     p1.div(5);
48     pvector p2 = pvector(1, 1);
49     BOOST_CHECK(p1 == p2);
50 }
```

### 7.71.3.4 BOOST\_AUTO\_TEST\_CASE() [4/12]

```
BOOST_AUTO_TEST_CASE (
    slt4 )
```

pvector dotproduct test case

Definition at line 55 of file test\_suites.cpp.

```
56 {
57     pvector p1 = pvector(1, 4);
58     pvector p2 = pvector(3, 2);
59     float dotProduct = p1.dotProduct(p2);
60     BOOST_CHECK(dotProduct == 11);
61 }
```

**7.71.3.5 BOOST\_AUTO\_TEST\_CASE()** [5/12]

```
BOOST_AUTO_TEST_CASE (
    slt5 )
```

pvector angle between vectors test case

Definition at line 66 of file test\_suites.cpp.

```
67 {
68     pvector p1 = pvector(10, 10);
69     pvector p2 = pvector(0, 10);
70     float angle = p1.angleBetween(p2);
71     BOOST_CHECK(angle == 45);
72 }
```

**7.71.3.6 BOOST\_AUTO\_TEST\_CASE()** [6/12]

```
BOOST_AUTO_TEST_CASE (
    slt6 )
```

pvector get vector angle test case

Definition at line 77 of file test\_suites.cpp.

```
78 {
79     pvector p1 = pvector(3, 4);
80     float angle = p1.getAngle();
81     BOOST_CHECK(angle < 53.2 && angle > 52.8);
82 }
```

**7.71.3.7 BOOST\_AUTO\_TEST\_CASE()** [7/12]

```
BOOST_AUTO_TEST_CASE (
    slt7 )
```

pvector normalize test case

Definition at line 87 of file test\_suites.cpp.

```
88 {
89     pvector p1 = pvector(2, 2);
90     p1.normalize();
91     float range = 0.01;
92     BOOST_CHECK_CLOSE_FRACTION(0.707, p1.x, range);
93     BOOST_CHECK_CLOSE_FRACTION(0.707, p1.y, range);
94 }
```

**7.71.3.8 BOOST\_AUTO\_TEST\_CASE()** [8/12]

```
BOOST_AUTO_TEST_CASE (
    slt8 )
```

pvector limit test case

Definition at line 99 of file test\_suites.cpp.

```
100 {
101     pvector p1 = pvector(2, 2);
102     p1.limit(3);
103     float range = 0.01;
104     BOOST_CHECK_CLOSE_FRACTION(2.12, p1.x, range);
105     BOOST_CHECK_CLOSE_FRACTION(2.12, p1.y, range);
106 }
```



### 7.71.3.9 BOOST\_AUTO\_TEST\_CASE() [9/12]

```
BOOST_AUTO_TEST_CASE (
    s1t9 )
```

pvector overloaded operators test case

Definition at line 111 of file test\_suites.cpp.

```
112 {
113     pvector p1 = pvector(1, 1);
114     p1 += pvector(1,1);
115     BOOST_CHECK(p1 == pvector(2,2));
116     p1 = pvector(1,1) + pvector(3,3);
117     BOOST_CHECK(p1 == pvector(4,4));
118     p1 = pvector(4,1) - pvector(3,3);
119     BOOST_CHECK(p1 == pvector(1,-2));
120     p1 = pvector(4,1) - point(3,3);
121     BOOST_CHECK(p1 == pvector(1,-2));
122     p1 = pvector(4,1) + point(3,3);
123     BOOST_CHECK(p1 == pvector(7,4));
124 }
```

### 7.71.3.10 BOOST\_AUTO\_TEST\_CASE() [10/12]

```
BOOST_AUTO_TEST_CASE (
    s2t1 )
```

point multiplication test case

Definition at line 133 of file test\_suites.cpp.

```
134 {
135     point p1 = point(1, 1);
136     p1.mul(3);
137     point p2 = point(3, 3);
138     BOOST_CHECK(p1 == p2);
139 }
```

### 7.71.3.11 BOOST\_AUTO\_TEST\_CASE() [11/12]

```
BOOST_AUTO_TEST_CASE (
    s2t2 )
```

point division test case

Definition at line 144 of file test\_suites.cpp.

```
145 {
146     point p1 = point(4, 4);
147     p1.div(4);
148     point p2 = point(1, 1);
149     BOOST_CHECK(p1 == p2);
150 }
```

### 7.71.3.12 BOOST\_AUTO\_TEST\_CASE() [12/12]

```
BOOST_AUTO_TEST_CASE (
    s2t3 )
```

point overloaded operators test case

Definition at line 155 of file test\_suites.cpp.

```
156 {
157     point p1 = point(1,1) + point(3,3);
158     BOOST_CHECK(p1 == point(4,4));
159     p1 = point(1,1) + pvector(3,3);
160     BOOST_CHECK(p1 == point(4,4));
161     pvector p2 = point(1,1) - point(3,3);
162     BOOST_CHECK(p2 == pvector(-2,-2));
163 }
```

## 7.72 /home/user/Desktop/mmm/autonomousSteeringAgents/wander.d File Reference

## 7.73 /home/user/Desktop/mmm/autonomousSteeringAgents/windy.d File Reference

# Index

/home/user/Desktop/mmm/autonomousSteeringAgents/README.md, 16  
134 /home/user/Desktop/mmm/autonomousSteeringAgents/include/pursuit.h,  
/home/user/Desktop/mmm/autonomousSteeringAgents/agent.d, 117  
93 /home/user/Desktop/mmm/autonomousSteeringAgents/include/pvector.h,  
/home/user/Desktop/mmm/autonomousSteeringAgents/color.d, 119  
93 /home/user/Desktop/mmm/autonomousSteeringAgents/include/random.h,  
/home/user/Desktop/mmm/autonomousSteeringAgents/entity.d, 120  
93 /home/user/Desktop/mmm/autonomousSteeringAgents/include/scenario.h,  
/home/user/Desktop/mmm/autonomousSteeringAgents/evade.d, 121  
93 /home/user/Desktop/mmm/autonomousSteeringAgents/include/steeringBe  
/home/user/Desktop/mmm/autonomousSteeringAgents/flee.d, 122  
93 /home/user/Desktop/mmm/autonomousSteeringAgents/include/wander.h,  
/home/user/Desktop/mmm/autonomousSteeringAgents/flock.d, 126  
93 /home/user/Desktop/mmm/autonomousSteeringAgents/include/windy.h,  
/home/user/Desktop/mmm/autonomousSteeringAgents/flowField.d, 128  
93 /home/user/Desktop/mmm/autonomousSteeringAgents/leaderFollower.d,  
/home/user/Desktop/mmm/autonomousSteeringAgents/graphics.d, 129  
93 /home/user/Desktop/mmm/autonomousSteeringAgents/main.cpp,  
/home/user/Desktop/mmm/autonomousSteeringAgents/include/agent.h, 129  
93 /home/user/Desktop/mmm/autonomousSteeringAgents/main.d,  
/home/user/Desktop/mmm/autonomousSteeringAgents/include/color.h, 134  
95 /home/user/Desktop/mmm/autonomousSteeringAgents/mouseFollower.d,  
/home/user/Desktop/mmm/autonomousSteeringAgents/include/entity.h, 134  
97 /home/user/Desktop/mmm/autonomousSteeringAgents/obstacle.d,  
/home/user/Desktop/mmm/autonomousSteeringAgents/include/evade.h, 134  
98 /home/user/Desktop/mmm/autonomousSteeringAgents/obstacleAvoidance  
/home/user/Desktop/mmm/autonomousSteeringAgents/include/flee.h, 134  
99 /home/user/Desktop/mmm/autonomousSteeringAgents/path.d,  
/home/user/Desktop/mmm/autonomousSteeringAgents/include/flock.h, 134  
101 /home/user/Desktop/mmm/autonomousSteeringAgents/pathFollower.d,  
/home/user/Desktop/mmm/autonomousSteeringAgents/include/flowField.h,  
102 /home/user/Desktop/mmm/autonomousSteeringAgents/point.d,  
/home/user/Desktop/mmm/autonomousSteeringAgents/include/graph.h, 134  
105 /home/user/Desktop/mmm/autonomousSteeringAgents/prison.d,  
/home/user/Desktop/mmm/autonomousSteeringAgents/include/leaderFollower.h,  
107 /home/user/Desktop/mmm/autonomousSteeringAgents/pursuit.d,  
/home/user/Desktop/mmm/autonomousSteeringAgents/include/mouseFollower.h,  
108 /home/user/Desktop/mmm/autonomousSteeringAgents/pvector.d,  
/home/user/Desktop/mmm/autonomousSteeringAgents/include/obstacle.h,  
109 /home/user/Desktop/mmm/autonomousSteeringAgents/random.d,  
/home/user/Desktop/mmm/autonomousSteeringAgents/include/obstacleAvoidance.h,  
111 /home/user/Desktop/mmm/autonomousSteeringAgents/scenario.d,  
/home/user/Desktop/mmm/autonomousSteeringAgents/include/path.h, 134  
112 /home/user/Desktop/mmm/autonomousSteeringAgents/src/agent.cpp,  
/home/user/Desktop/mmm/autonomousSteeringAgents/include/pathFollower.h,  
113 /home/user/Desktop/mmm/autonomousSteeringAgents/src/color.cpp,  
/home/user/Desktop/mmm/autonomousSteeringAgents/include/point.h, 135  
115 /home/user/Desktop/mmm/autonomousSteeringAgents/src/entity.cpp,  
/home/user/Desktop/mmm/autonomousSteeringAgents/include/prison.h, 136



- G, [23](#)
- getColor, [23](#)
- R, [23](#)
- color.h
  - BLACK, [96](#)
  - BLUE, [96](#)
  - CYAN, [96](#)
  - GREEN, [96](#)
  - MAGENDA, [96](#)
  - RED, [97](#)
  - WHITE, [97](#)
  - YELLOW, [97](#)
- createObstacle
  - obstacleAvoidance, [50](#)
- createPath
  - pathFollower, [55](#)
- createRandomAgents
  - scenario, [76](#)
- createRandomArray
  - random, [74](#)
- createStaticAgents
  - scenario, [77](#)
- createTroop
  - scenario, [77](#)
- CYAN
  - color.h, [96](#)
- desiredVelocity
  - agent, [19](#)
- DISTANCE
  - prison.cpp, [151](#)
- div
  - point, [58](#)
  - pvector, [68](#)
- dotProduct
  - pvector, [68](#)
- draw
  - agent, [15](#)
  - entity, [25](#)
  - obstacle, [47](#)
  - path, [53](#)
- drawAgent
  - graphics, [34](#)
- drawCircle
  - graphics, [35](#)
- drawLine
  - graphics, [35](#)
- drawPath
  - graphics, [36](#)
- drawPoint
  - graphics, [36](#)
- drawText
  - graphics, [36](#)
- entity, [24](#)
  - draw, [25](#)
  - entity, [24](#)
  - getColor, [25](#)
  - getId, [25](#)
  - getName, [25](#)
  - setColor, [26](#)
  - setId, [26](#)
  - setName, [26](#)
- ESC
  - graphics.h, [106](#)
- EVADE
  - steeringBehavior.h, [124](#)
- evade, [27](#)
  - evade, [27](#)
  - loop, [28](#)
  - steeringBehavior, [83](#)
- FIELD\_HEIGHT
  - flowField.h, [104](#)
- FIELD\_WIDTH
  - flowField.h, [104](#)
- FLLEE
  - steeringBehavior.h, [124](#)
- flee, [28](#)
  - flee, [29](#)
  - loop, [29](#)
  - steeringBehavior, [83](#)
- FLOCK
  - steeringBehavior.h, [125](#)
- flock, [30](#)
  - flock, [30](#)
  - loop, [30](#)
- flow
  - windy, [91](#)
- flowField, [31](#)
  - flowField, [31](#), [32](#)
  - getField, [32](#)
- flowField.h
  - FIELD\_HEIGHT, [104](#)
  - FIELD\_WIDTH, [104](#)
  - GRAVITY, [104](#)
  - WIND\_WEST, [104](#)
- FOLLOW\_MOUSE
  - steeringBehavior.h, [125](#)
- force
  - agent, [19](#)
- forceInScreen
  - graphics, [37](#)
- G
  - color, [23](#)
- getAngle
  - pvector, [69](#)
- getCenter
  - obstacle, [48](#)
- getColor
  - color, [23](#)
  - entity, [25](#)
- getField
  - flowField, [32](#)
- getId
  - entity, [25](#)
- getMass

- agent, 16
- getMousePosition
  - graphics, 37
- getName
  - agent, 16
  - entity, 25
- getNormalPoint
  - point, 59
- getPathWidth
  - path, 53
- getRadius
  - obstacle, 48
- getTarget
  - agent, 16
- getVelocity
  - agent, 16
- graphics, 33
  - drawAgent, 34
  - drawCircle, 35
  - drawLine, 35
  - drawPath, 36
  - drawPoint, 36
  - drawText, 36
  - forceInScreen, 37
  - getMousePosition, 37
  - handleKeypress, 37
  - handleResize, 38
  - initGraphics, 38
  - mouseButton, 40
  - mouseMove, 40
  - refreshScene, 41
  - target\_x, 41
  - target\_y, 42
  - timerEvent, 41
- graphics.h
  - ESC, 106
  - HEIGHT, 106
  - PI, 106
  - WIDTH, 106
- GRAVITY
  - flowField.h, 104
- GREEN
  - color.h, 96
- handleKeypress
  - graphics, 37
- handleResize
  - graphics, 38
- HEIGHT
  - graphics.h, 106
- id
  - agent, 20
- IN\_FLOW\_FIELD
  - steeringBehavior.h, 125
- inFlowField
  - steeringBehavior, 84
- initGL
  - scenario, 78
- initGraphics
  - graphics, 38
- LEADER\_FOLLOWER
  - steeringBehavior.h, 125
- leaderFollower, 42
  - leaderFollower, 43
  - leaderPosition, 44
  - leaderVelocity, 44
  - loop, 43
- leaderPosition
  - leaderFollower, 44
- leaderVelocity
  - leaderFollower, 44
- limit
  - pvector, 69
- loop
  - evade, 28
  - flee, 29
  - flock, 30
  - leaderFollower, 43
  - mouseFollower, 45
  - obstacleAvoidance, 50
  - pathFollower, 56
  - prison, 63
  - pursuit, 64
  - wander, 89
  - windy, 91
- MAGENDA
  - color.h, 96
- magnitude
  - pvector, 69
- main
  - main.cpp, 130
- main.cpp
  - main, 130
  - menu, 131
  - mode, 132
- MAX\_NUMBER\_OF\_AGENTS
  - scenario.cpp, 155
- maxForce
  - agent, 20
- maxSpeed
  - agent, 20
- menu
  - main.cpp, 131
- mode
  - main.cpp, 132
- mouseButton
  - graphics, 40
- mouseFollower, 45
  - loop, 45
  - mouseFollower, 45
- mouseMove
  - graphics, 40
- mul
  - point, 59
  - pvector, 69

- myPath
  - pathFollower, 56
- name
  - scenario, 79
- normalize
  - pvector, 70
- obstacle, 46
  - draw, 47
  - getCenter, 48
  - getRadius, 48
  - obstacle, 47
  - setCenter, 48
  - setRadius, 48
- obstacleAvoidance, 49
  - createObstacle, 50
  - loop, 50
  - obstacleAvoidance, 49
  - obstacles, 51
- obstacles
  - obstacleAvoidance, 51
- operator+
  - point, 60
  - pvector, 70, 71
- operator+=
  - pvector, 71
- operator-
  - point, 60
  - pvector, 72
- operator==
  - point, 61
  - pvector, 72
- path, 51
  - addPoint, 53
  - draw, 53
  - getPathWidth, 53
  - path, 52
  - points, 54
  - setPathWidth, 54
- pathFollower, 54
  - createPath, 55
  - loop, 56
  - myPath, 56
  - pathFollower, 55
- PI
  - graphics.h, 106
  - pvector.h, 120
- point, 57
  - div, 58
  - getNormalPoint, 59
  - mul, 59
  - operator+, 60
  - operator-, 60
  - operator==, 61
  - point, 57, 58
  - print, 61
  - x, 62
  - y, 62
- points
  - path, 54
- position
  - agent, 20
- print
  - point, 61
  - pvector, 73
- prison, 62
  - loop, 63
  - prison, 63
- prison.cpp
  - DISTANCE, 151
  - WALL, 151
- PURSUIT
  - steeringBehavior.h, 125
- pursuit, 64
  - loop, 64
  - pursuit, 64
  - steeringBehavior, 85
- pvector, 65
  - add, 67
  - angleBetween, 67
  - div, 68
  - dotProduct, 68
  - getAngle, 69
  - limit, 69
  - magnitude, 69
  - mul, 69
  - normalize, 70
  - operator+, 70, 71
  - operator+=, 71
  - operator-, 72
  - operator==, 72
  - print, 73
  - pvector, 66
  - x, 73
  - y, 73
- pvector.h
  - PI, 120
- R
  - color, 23
- r
  - agent, 20
- RANDOM
  - scenario.h, 122
- random, 74
  - createRandomArray, 74
- RED
  - color.h, 97
- refresh
  - scenario, 78
- refreshScene
  - graphics, 41
- scenario, 75
  - agents, 79
  - behavior, 79

- callback, 79
- createRandomAgents, 76
- createStaticAgents, 77
- createTroop, 77
- initGL, 78
- name, 79
- refresh, 78
- scenario, 76
- view, 79
- scenario.cpp
  - MAX\_NUMBER\_OF\_AGENTS, 155
- scenario.h
  - RANDOM, 122
  - STATIC, 122
  - TROOP, 122
  - types, 122
- seek
  - steeringBehavior, 85
- separation
  - steeringBehavior, 86
- setCenter
  - obstacle, 48
- setColor
  - entity, 26
- setFeatures
  - agent, 16
- setId
  - entity, 26
- setMass
  - agent, 17
- setName
  - agent, 17
  - entity, 26
- setPathWidth
  - path, 54
- setRadius
  - obstacle, 48
- setTarget
  - agent, 17
- setVelocity
  - agent, 18
- STATIC
  - scenario.h, 122
- STAY\_IN\_FIELD
  - steeringBehavior.h, 125
- STAY\_IN\_PATH
  - steeringBehavior.h, 126
- stayInArea
  - steeringBehavior, 86
- stayInPath
  - steeringBehavior, 87
- steering
  - agent, 21
- steeringBehavior, 80
  - align, 81
  - avoid, 81
  - cohesion, 82
  - evade, 83
  - flee, 83
  - inFlowField, 84
  - pursuit, 85
  - seek, 85
  - separation, 86
  - stayInArea, 86
  - stayInPath, 87
  - wander, 88
- steeringBehavior.h
  - AVOID\_OBSTACLE, 124
  - CIRCLE\_DISTANCE, 124
  - CIRCLE\_RADIUS, 124
  - EVADE, 124
  - FLEE, 124
  - FLOCK, 125
  - FOLLOW\_MOUSE, 125
  - IN\_FLOW\_FIELD, 125
  - LEADER\_FOLLOWER, 125
  - PURSUIT, 125
  - STAY\_IN\_FIELD, 125
  - STAY\_IN\_PATH, 126
  - WANDER, 126
- target\_x
  - graphics, 41
- target\_y
  - graphics, 42
- test\_suites.cpp
  - BOOST\_AUTO\_TEST\_CASE, 160–163
  - BOOST\_TEST\_MODULE, 160
- timerEvent
  - graphics, 41
- TROOP
  - scenario.h, 122
- types
  - scenario.h, 122
- updatePosition
  - agent, 18
- view
  - scenario, 79
- WALL
  - prison.cpp, 151
- WANDER
  - steeringBehavior.h, 126
- wander, 89
  - loop, 89
  - steeringBehavior, 88
  - wander, 89
- WHITE
  - color.h, 97
- WIDTH
  - graphics.h, 106
- WIND\_WEST
  - flowField.h, 104
- windy, 90
  - flow, 91



loop, [91](#)  
windy, [91](#)

x

point, [62](#)  
pvector, [73](#)

y

point, [62](#)  
pvector, [73](#)

YELLOW

color.h, [97](#)