Function	**Information**
From Game	
getWaypoints()	Returns the list with all the waypoints in the map.
getWaypointsLeft()	Indicates the number of waypoints that are still to be visited.
getWaypointsVisited()	Returns the number of waypoints visited in this play.
getFuelTanks()	Returns the list with all the fuel tanks in the map.
getFuelTanksLeft()	Indicates the number of fuel tanks that have been not collected yet.
getFuelTanksVisited()	Returns the number of waypoints colected so far.
getNumFuelTanks()	Returns the total number of fuel tanks in the game.
getTotalTime()	Gets the time spent since the beginning of the game.
getStepsLeft()	Returns the steps left until the time runs out.
getMap()	Returns the map of the game (instance of Map).
getMapSize()	Returns the dimensions of the map (instance of java.awt.Dimension).
getShip()	Gets the ship of the game (instance of class Ship).
isEnded()	Indicates if the game is ended.
advanceMap()	Advances the current map to the next loaded one.
getVisitOrder()	Returns the visit order of the game so far.
getCopy()	Gets a copy of the whole game state in a Game object.
From Waypoint	
isCollected()	Indicates if this waypoint has been collected/visited.
Vector2d s	Indicates the position of this waypoint.
RADIUS	Represents the radius (number of pixels) of the waypoints.
From FuelTank	
isCollected()	Indicates if this fuel tank has been collected.
Vector2d s	Indicates the position of this fuel tank.
RADIUS	Represents the radius (number of pixels) of the fuel tank.
From Map	
getMapChar()	Gets a bi-dimensional array with the contents of the current map. Each position is a pixel on the map, and a character of the map file.
getMapHeight()	Gets the height of the map (in pixels).
getMapWidth()	Returns the width of the map (in pixels).
getStartingPoint()	Gets the starting point of the ship.
isObstacle(x,y)	Returns true if there is an obstacle in the position given.
isLava(x,y)	Indicates if a given position in the grid is of a lava surface.
LineOfSight(origin, destination)	Checks if there are no obstacles from the origin position to the destination (considering ship radius).
distToCollision(v,w,d)	Returns the distance to a potential obstacle from a given point (v), in a specified direction (w) an up to a maximum distance (d). Gets -1 if no collision.
getCopy()	Gets a copy of the Map object.

TABLE I: Code interface I.

Function	**Information**
From Ship	
getCollLastStep ()	Indicates if there was a collision in the last step.
getLastCollisionType ()	Returns the type of the last collision
checkCollisionInPosition (pos)	Checks if there is a collision in *pos*, considering the ship's bounding sphere.
getCollisionTypeInPosition (pos)	Checks the type of collision in a position given by Vector2d *pos*.
getInvulnerableTime ()	Returns the remaining invulnerable time.
getDamage()	Returns the damage of the ship.
getRemainingFuel()	Returns the remaining fuel in the ship.
isOnLava()	Indicates if the ship is on a lava surface.
update(action)	Performs the action provided.
getCopy()	Gets a copy of the Ship object.
Vector2d s	Position of the ship.
Vector2d sp	Position of the ship in the previous step.
Vector2d v	Velocity of the ship.
Vector2d d	Direction of the ship (where the ship is facing, not necessarily the same as Velocity).
SHIP_RADIUS	Represents the radius (number of pixels) of the ship.
From Controller (static methods)	
getThrust (action)	Returns true if the action given accelerates the ship.
getTurning (action)	Returns -1, 1 or 0 if the action given rotates left, right or none, respectively.
getActionFromInput(thrust , turn)	Given an acceleration boolean and a turn sense, returns the desired action identifier.

TABLE II: Code interface II.

Constant	**Information**
PTSPConstants.DELAY	Delay in milliseconds between screenshots (used for replays and human plays).
PTSPConstants.T	Physics time.
PTSPConstants.STEPS_PER_WAYPOINT	Number of steps allowed until reaching the next waypoint.
PTSPConstants.NO_COLLISION	Collision type: no collision.
PTSPConstants.NORMAL_COLLISION_TYPE	Collision type: normal collision.
PTSPConstants.DAMAGE_COLLISION_TYPE	Collision type: extra damaging collision.
PTSPConstants.ELASTIC_COLLISION_TYPE	Collision type: elastic collision.
PTSPConstants. DAMAGE_NORMAL_COLLISION	Damage suffered by the ship when colliding with a normal collision
PTSPConstants. DAMAGE_DAMAGE_COLLISION	Damage suffered by the ship when colliding with a damage collision.
PTSPConstants.DAMAGE_LAVA	Damage suffered by the ship when being on a lava surface.
PTSPConstants.COLLISION_SPEED_RED	The velocity of the ship will be multiplied by this amount when colliding with a normal wall.
PTSPConstants. COLLISION_DAMAGE_SPEED_RED	The velocity of the ship will be multiplied by this amount when colliding with a DAMAGE wall.
PTSPConstants. COLLISION_ELASTIC_SPEED_RED	The velocity of the ship will be multiplied by this amount when colliding with a ELASTIC wall.
PTSPConstant.MAX_DAMAGE	Maximum damage the ship can hold before being destroyed.
PTSPConstant.INITIAL_FUEL	Initial (and maximum) fuel for the ship.
PTSPConstant.FUEL_TANK_BOOST	Amount of fuel gained when a fuel tank is collected.
PTSPConstant.FUEL_WAYPOINT_REWARD	Fuel reward for visiting a waypoint.
PTSPConstant.INVULNERABLE	Time the ship is set to invulnerable after a collision.
PTSPConstants.INIT_TIME_MS	Time for the controller to be initialized.
PTSPConstants.ACTION_TIME_MS	Time for the controller to provide an action every step.
PTSPConstants.TIME_ACTION_DISQ	If the controller spends more than TIME_ACTION_DISQ to reply with an action, it gets disqualified from this game (final score: 0 waypoints, PTSPConstants.getStepsPerWaypoints() time steps, 0 as remaining fuel and PTSPConstants.MAX_DAMAGE as damage).
PTSPConstants.getStepsPerWaypoints(nwp)	Returns the number of time steps until reaching the next waypoint. nwp is the number of waypoints of the map.
Controller .ACTION_NO_FRONT	Action: No thrust, no rotation.
Controller .ACTION_NO_LEFT	Action: No thrust, rotate left.
Controller .ACTION_NO_RIGHT	Action: No thrust, rotate right.
Controller .ACTION_THR_FRONT	Action: Thrust, no rotation.
Controller .ACTION_THR_LEFT	Action: Thrust, rotate left.
Controller .ACTION_THR_RIGHT	Action: Thrust, rotate right.
Controller .NUM_ACTIONS	Number of different actions that can be applied at each step.
Controller .HALF_PI	PI / 2
Controller .QUARTER_PI	PI / 4

TABLE III: Useful constants.