filed on: {indigo}<kbvlsi>loopscourse>truckinvocabulary.bravo
Last Edited: SM: July 11, 1983

Truckin Query Functions

by the Loops Design Team

Daniel Bobrow, Sanjay Mittal, and Mark Stefik

This document summarizes the functions and methods you will find useful in writing the rules for your *Truckin* players. These functions allow you to select and filter roadstops satisfying different constraints as well as conveniently access other information about the current status of the *Truckin* world. Many of the following functions are also available as methods attached to the class **Player**, allowing you to easily specialize them if you so desire.

In the following summary, functions marked with an asterisk (*) are also implemented as methods on **Player** with the same name as the function and taking the exact same arguments. For more details about these functions see the listing of the file TRUCKINV in your folder.

A. Selection functions

The following functions return a list of roadstops based on certain constraints.

AnyRoadStop (roadStopType numMoves direction roomToParkFlg)*
Randomly picks one of the roadstops of type *roadStopType* where *roadStopType* is one of the RoadStop classes. If *numMoves* is provided, it returns only those roadstops within that distance. If *direction* is F then only those in the forward direction, if B then only in the backward direction, if NIL then in either direction. If *roomToParkFlg* is T then only those roadstops where there is room to park.

Buyers (commodityClass numMoves includeCDFlg)*
Returns all of the Buyers (i.e. *Consumer* roadstops) able to purchase a commodity of type *commodityClass*. If *numMoves* is provided, returns only those within that distance. A common case is to use the instance variable *maxMove* of your player as this argument. If *includeCDFlg* is T then includes *CityDumps* also, otherwise not.

NthRoadStop (numMoves direction fromRoadStop roomToParkFlg)*
Returns the Nth roadstop in the given direction from *fromRoadStop*. If *fromRoadStop* is NIL, the current location of the player is used. If *direction* is NIL, **Forward** is assumed. If there are fewer than *numMoves* roadstops in the specified direction, that is if the request would go off the board, this function returns the farthest roadstop in that direction.

RoadStops (roadStopType numMoves direction roomToParkFlg)*
Returns all of the roadstops of type *roadStopType* reachable within *numMoves* in the direction specified by *direction* taking into account room to park if *roomToParkFlg* is T.

Sellers (commodityClass numMoves)*
Returns all the roadstops which are Sellers (i.e. *Producer* roadstops) of *commodityClass* and are located within *numMoves.*

B. Filter functions
The following functions take a set of roadstops as one argument and prune that set based on other criteria specified by other arguments. Some of the following functions are very general and can be used to filter (or order) any set of objects of the same class and are not limited to working on roadstops only. These are: **FilterObjs, PickHiObj, PickLowObj,** and **SortObjs.**

FilterObjs (self selector objects)
Sends a *selector* msg to *self* for each of the object in *objects* and returns all of the objects for which the rule set returned a non-NIL value. This is the basic function for doing filtering based on your knowledge encoded as rules.

FurthestRoadStop (roadStops fromRoadStop)*
Returns the roadstop in *roadStops* which is furthest from *fromRoadStop* excluding *fromRoadStop.* If *fromRoadStop* is NIL, assumes the current location of the player.

NearestRoadStop (roadStops fromRoadStop)*
Same as **FurthestRoadStop** except returns the nearest roadstop.

PickHiObj (self selector objects)
Sends a *selector* msg to *self* for each object in *objects* to determine a numeric rating for each of the objects. It returns the object with the highest numeric rating. When the value returned is non-numeric for an object, then that object is automatically excluded.

PickLowObj (self selector objects)
Same as **PickHiObj** except returns the one with the lowest numeric rating.

SortObjs (self selector objects)
Sends a *selector* msg to *self* for each object in *objects* to determine a numeric rating for each of them. It returns a list of objects in the **descending** order of their numeric rating. It also excludes the ones with non-numeric ratings.

C. Miscelleneous functions

AnyBanditsP (toRoadStop fromRoadStop)
Returns T if there are any bandits parked between *toRoadStop* and *fromRoadStop,* NIL otherwise.

DirectionOf (toRoadStop fromRoadStop)*
Returns the direction of travel for going from *fromRoadStop* to *toRoadStop.* If the *fromRoadStop* is not given, then the current location of the player is assumed.

Distance (toRoadStop fromRoadStop)*

Computes the distance between *fromRoadStop* and *toRoadStop*. If the *fromRoadStop* is not given, then the current location of the player is assumed.

PricePerUnit (producerRoadStop)
Returns the buying price per unit of the commodity being sold at the *producerRoadStop*. If the argument is not *a Producer* roadstop, then complains and returns 1.

RoomToParkP (roadStop)
Returns T if there is room to park at *roadStop*.

ISA (instance className)
Returns T if *instance* is an instance of *className*.

Nth (list index)
Returns the *index* element of *list*.

SUBCLASS (class superClass)
Returns T if *class* is same as or a subclass of *superClass*.

The following are available only as methods on **Player** class.

(← player Range)
Computes how far the *player* can move based on the amount of fuel carried on the player's truck.

(← player Range1)
Computes how far the *player* can move in a single turn. This depends on the fuel in the truck and the maximum distance allowed by the game master for that turn.

(← player TimeAtStop)
Returns the time spent by player at the stop where currently parked. Useful when parked at one of the Alice's.

(← player TurnsAtStop)
Returns the number of turns *player* has been parked at the stop where currently parked. Useful when parked at one of the Alice's.

D. Useful Global Variables

1. PlayerInterface (you can also use **PI**)
After doing (← $Truckin New), *PlayerInterface* is bound to the instance of the class *TruckinPlayerInterface* and is used to send messages to the GameMaster for making moves and starting game. You can also get some game information such as *roadStops* and *localPlayers* from this object.

2. Simulator

Once the game is set up, *Simulator* is bound to the instance of *TruckinSimulator* and can be used to access important game information such as *roadStops, players, beginTime, endTime, timeLeft.*

3. debugMode

If set to *T*, then each time a rule is violated, the *RuleExec* is automatically brought up. Useful while debugging your rulesets. If set to *NIL*, then the RuleExec is not entered for each rule violation. Also, the GameMaster traps all errors. Initially set to *T*.

4. truckinLogFlg

If set to *T*, before game is started, then prepares a log file of all important game messages in a file called *TRUCKINLOG*. This log file may be useful during the debugging of your players. Set this variable to NIL, if you dont want any log file. Initially set to NIL.