CS403: Introduction to Empirical Modelling
Car Simulator Model

Salman Shahid Butt
0961493
s.s.butt@warwick.ac.uk

Abstract

This model uses the principles of Empirical Modelling to make a car model. This model attempts to mimic a real world car, so that when turning the car it actually calculates the angle at which the car will turn based on the amount of turn. This thesis will show how easy it is to construct models using EM techniques, the importance of using EM techniques compared to traditional programming languages and the problems encountered in developing such a model. This makes use of EDEN interpreter and other definitive notations such as SCOUT and DONALD.

1 Introduction

This paper is concerned with making a model of a car which hasnt been done using EM principles in the past. The EM project archive**?**) contains few of models related to cars , such as Car Parking Simulator**?**) and Racing Cars**?**). This model is different in a sense that it tries to show the actual angle of turn each of the wheels take while tuning the vehicle. For example it will show the point at which the car will turn based on the turning of the wheels. Furthermore the users have the ability to turn and move the car using arrow keys making is seem like a more realistic experience. This model can be further extended to show how the car turning will vary if for example the car was towing a caravan or if it were a lorry. The turning angles will be different for all. For a lorry, larger turns will have to be made in order successfully make a turn and so on.

2 Empirical Modelling

Empirical Modelling is different from traditional programming. It provides a radically new approach to developing software. EM is a collection of tools and principles concerned with modelling state. It is an approach to constructing computer based models. Its empirical in a sense that the user develops the model through experimenting and observing the real world. Dr. Beynon mentions**?**). EM entails the development of construals; interactive artefacts that embody the learners personal understanding of a situation and referent. Such construals are developed by imitating the relations between observables, dependencies and agent actions that are identified as characteristic of the referent. EM uses a set of definitive notations such as DONALD which is used for 2d line drawing, SCOUT, used to display windows onto the screen, ARCA, used to display and describe combinatorial graphs and more. All these tools have been developed at the University of Warwick; the tools are constantly being improved, the newest addition being DOSTE. The notations used in this project are DONALD for drawing the car and SCOUT to display the car in a window. It makes use of EDEN interpreter. DOSTE is the most recent notation, but was not used for modelling because there was not enough documentation available.

3 Modelling the Car

3.1 Initial Design

It may be surprising to some that when turning the car the front wheels are not pointing in the same direction. Each wheel must follow a different circle. The inside wheel is following a circle with a smaller radius than the outside wheel. If perpendicular lines are drawn from each wheel, they will intersect at the centre point of the turn. The following example shows the turning of the car.

The model was made with this example in mind. Initially simple observables and its dependencies were determined, the car itself was a rectangle drawn using DONALD which used two points to draw a rectangle. It was also thought that the agent will be

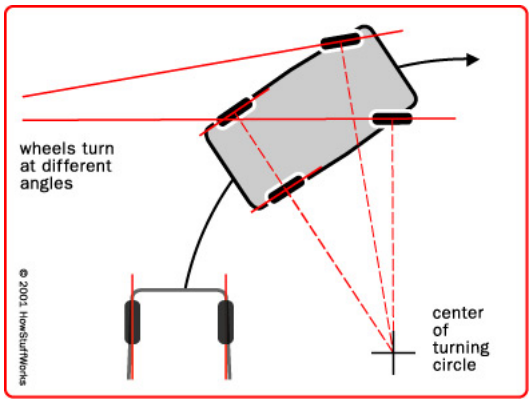

Figure 1: A Figure showing the Car turn. Source howstuffworks.com

the users who can use the arrow keys to drive it like for example in game.

```
%donald
/* observables defining
the shape of the car
used to build the rectangle
 shape in Donald*/
point pcar, pcar1
rectangle rec
int x,y
x=425
y = 300
pcar = {x,y}
pcar1 = {x+80,y+200}
/*rectangle drawn which
 signifies the
 shape of the car*/
rec= rectangle(pcar, pcar1)
```

As the model progressed further observables relating to the car were establishes as they would be needed to make the car behave as the modeller intended. Such observables are for example the midpoint of the car, the wheels and its midpoints. It was also helpful to make figure out the length and width of the car and also the midpoints of the front and back of the car. An observable was set up to determine the direction of the car rightly named cardir. The following shows the new observables and dependencies

```
/*New observables for
the length, width and
 cardir defined along
with the midpoints*/
real length, width
real cardir
point midfront, midrear
line caraxis
/*the midpoints of front of the
 car and the back are calculated*/
midfront = midcar + {length div 2@cardir}
midrear = midcar - {length div 2@cardir}
/*the caraxis is a line
 which will join the midpoints */
caraxis = [midfront, midrear]
/*dependencies established
between the two points
which define the rectangle.
 It will be dependant on the observable
  we defined earlier cardir.*/
pcar = midrear + {width@(cardir + pi div 2)}
pcar1 = midfront - {width@(cardir + pi div 2)}
```

At this point simple dependencies were set up using the cardir observable and the idea was to rotate the car using that observable. As seen from the diagram the car turns around at turning centre which is determined by the intersection of the perpendicular lines drawn from the wheels of the car. A new observable which should reflect this was set up called turncentre. By moving the turncentre up and down the angle of the turn will be determined. What this means is that if the turn centre is further away from the car itself the angles will be smaller, meaning the radius of the circle the car turns around would be larger. Similarly if turncentre is closer to the car the circle radius will be smaller and the car will make a smaller turn. This is an important step to determine how the car should turn sensibly. This was proving a bit of a challenge and was solved with the help of Dr. Beynon.
The wheels in the model are just lines drawn with DONALD. The back wheels remain invisible due to the fact that they overlap on the frame of the car. The front wheels are only allowed to move as it should be. The perpendicular lines connecting wheels were connected at turncentre. These lines are connected from their centres and there is an observable defined called steerangle which is dependent upon the angle of cardir, which was defined earlier.
The next step was to calculate the angles between the two lines connecting the midpoint of the front wheels. This was made possible again thanks to Dr. Beynon. It calculates the angles using arctan, between both lines connecting turncentre and midpoints of front wheels.

```
real oangle, nangle
```

```
oangle = atan((centreobw.1 - centreofw.1)
 div (centreobw.2 - turncentre.2))
nangle = atan((centrenbw.1 - centrenfw.1)
 div (centrenbw.2 - turncentre.2))
```

After this there is no need to change steerangle as the angle of the turn would be determined by changing turncentre for turning. The other observable needed to make the car move would be to change the midpoint of the car.

3.2 Problems Encountered

This model was not free from errors. The errors had to be constantly fixed but the real flaw which was later discovered was concerned with how DONALD manipulates the rectangle. As mentioned before the two observables needed to move the car and to steer the car were midcar and cardir respectively. It was discovered that changing midcar which is a DONALD point can be updated by changing the x and y coordinates which worked perfectly but problems arose when steering i.e changing the cardir observable. The problem was that DONALD calculates the rectangle based on two input parameters which are points with x and y values. It was assumed that changing the points which are dependent upon cardir should change and recalculate the rectangle at the points. When changing cardir it actually reshaped the model and it did not make sense why that would happen.
After experimenting for a long time it was discovered that the rectangle cannot be drawn as expected. DONALD did not have enough parameters to control the movement of the rectangle so the idea was scrapped and replaced with horizontal and vertical lines connected together to form a rectangle car shape.

3.3 Final Model

As mentioned earlier the rectangle method of DONALD was insufficient to make the model as expected. The idea was changed in favour of simple lines connected to each other to form a rectangle. After which the model was working fine and could be rotated about the cardir and moved by changing midcar. Its still not perfect, there is lot of things which should be corrected and some functionality added for it to simulate a real car.

3.4 Further Work

The model of the car itself can be enhanced to make it look more like a car, the car so far is just some lines connected with each other to form a rectangle. Since its a top down view of the car, the wheels cannot be shown too clearly. Only the turning can be shown graphically by changing the angle of the wheels. The colours could also be added to make it more visually attractive. The wheels at the moment can turn in any direction which should not be the case. There should be constraints put on the angle because the wheels should not be able to turn above or below certain angles for example the car cannot turn 90 degrees. These constraints are important to simulate the real world example. The car should ideally be controlled by arrow keys to make it rotate and move or there should be some buttons on the screen which interacts with the mouse to make it move. This sort of agency should be present; so far the observables can only be manipulated by using the tkeden input. To further enhance it there should be roads so the user can drive through them and see how the car behaves while turning. Next step is to introduce different types of vehicles. This will be an important study to see how different vehicles behave while turning. For example, a lorry will have to take a longer turn compared to car to make a successful turn otherwise it will crash or go off road. The main tool used was the EDEN interpreter along with DONALD and SCOUT.

4 Benefits of EM

This thesis showed how the development of a car model using the EM tools turned out. There were some problems in the beginning due to the limitations of DONALD, which were later solved. Nevertheless with the correct understanding of the EM principles and good knowledge of tools one can easily model and instantly visualize the results. Unlike traditional programming the EM construals are always running which means that redefinitions can be made at any time and the results will be instantly updated. As the modeller gains more knowledge of the artefact he can start redefining and updating the model. This in a sense is not very constrained as traditional programming languages but it also means that the modeller has to be careful and keep track of the dependencies otherwise odd errors may occur. The other important feature its very mathematical in nature and the benefit is that mathematical language offers a precise and clear representation of the problem situation relation to phenomena in the world. For example in DONALD the trigonometric formulas were required to calculate the angle at which the car will turn which reflects the real world model. It would have been very hard to visualise this in any other languages. The concept EM provides in modelling compared with pro-

gramming is unique and gives the ability to directly experience the phenomena rather than abstractions used in traditional programming practises. However there is still a need to build better tools with more functionality. For example tools with a graphical interface for DONALD would be a good addition. It will further the ability to visualize the objects and change them accordingly.

5 Conclusions

EM is a new approach to computing. The central activity of EM is building artefacts (models) which embody the modeller's personal construal of a phenomenon and consist of collections of observables, dependencies and agents. The main tool provided by EM the EDEN interpreter which can interact with many definitive notations used for modelling state. A car model was made using EM tools, which shows the steering of the car and how the angles of the wheels change. There were some problems encountered due to the shortcomings of DONALD for example, but were fixed by using the alternatives. The model is still incomplete, but it shows how the principles of EM can be used to model. Overall EM is very well suited to this study of car modelling, though not complete it is open to vast improvement.

6 Acknowledgements

I would like to thank Dr. Meurig Beynon for his tremendous effort in helping me model this car.

7 References