

Team 3

Optimization of roundabouts

Laura Fioretti

Fabio Piacentini

UNIVERSITA' DEGLI STUDI DI MILANO

Lei Liu

Technische Universität Kaiserslautern,

Robert Setekera

Xijian Wang

Eindhoven University of Technology

Instructor: Jan Homeyer

Technische Universität Kaiserslautern

1 Preface

This report about optimization of roundabouts is one of the problems during 22nd ECMI modeling week. It is the result of collaboration between five students from Three different universities, namely: UNIVERSITA' DEGLI STUDI DI MILANO, Technische Universitat Kaiserslautern and Eindhoven University of Technology. During the modeling week which took place at Eindhoven University of Technology (The Netherlands) from 17th August to 24th August in 2008, we got helpful comments and interesting suggestions from Dipl.-Math. Jan Homeyer from Technische Universitat Kaiserslautern , for which we are so thankful.

Laura Fioretti
Lei Liu
Fabio Piacentini
Robert Setekera
Xijian Wang

August 2008

2 Introduction

A lot of cities are recently replacing crossroads with high traffic volume by roundabouts (see Figure 1 below). These traffic circles supposed to reduce traffic jams and to simplify left turns (or right turns in Great Britain). In general these roundabouts have to be fitted into a given road network. The difficulties thereby are for example existing buildings and of course the limited space.

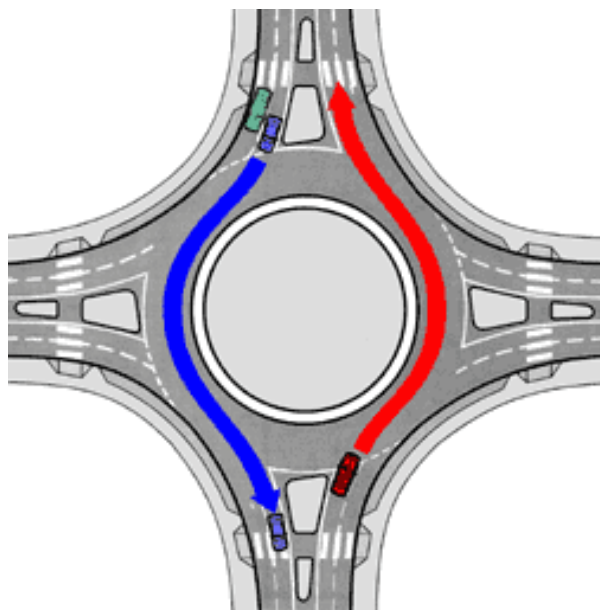


Figure 1: Roundabout

Now what are the problems the vehicles encounter when driving through a roundabout?. Often you can observe that large vehicles, for example eighteen-wheeled trucks or cars with trailers, accidentally also drive on the sidewalk (which should only be used by cyclists), and not only on the road. Often the effect that the rear wheels of the vehicle sheers out of line can in general not be completely compensated by steering.

A driving hazard problem was proposed by Baylis [1] and revised by Bender [2]. The problem can be described as follows: when making a right turn on British roadways, one moves to the right as far as possible on one's own side of the roadway and then turns. Unfortunately, the rear of the vehicle swings leftward as the right turn is begun - toward the unsuspecting driver passing on the left. This can be quite noticeable if the turning vehicle is a long bus. Hence the driving hazard problem - just how far will the back corner of the bus swing toward the left as the bus driver negotiates the right turn - was proposed. A similar problem, studied by Freedman and Riemenschneider [3], is how to determine the path of the rear wheel(s) of a bus while the front wheel of the bus tracks a given path.

In this article, through simulating the process of moving into and moving out of a roundabout of a vehicle(only cars and buses are considered, no trucks) by Matlab, we try to find the optimal size(minimization size) and shape of roundabouts.

3 Mathematical Model for solving the problem

3.1 Method 1

In this model we consider as given the path of the front wheels and the length of the vehicle. The rear wheels are supposed to move every time on the direction given by the vector between the front middle point of the wheels and the rear one.

If we consider $\underline{x}(t)$ as the path of the front point of the front wheel and $\underline{y}(t)$ as the one of the rear wheels whose distance from the front one is l , we have the following system of equation:

$$\begin{cases} \|\underline{x}(t) - \underline{y}(t)\| = l \\ \dot{\underline{y}}(t) \times (\underline{x}(t) - \underline{y}(t)) = 0 \end{cases}$$

Here x and y are the middle points of the axes respectively between the front wheels and rear wheels, $\dot{y}(t)$ is the velocity of the wheels, and norm we used is the Euclidean norm ($2 - norm$). The implementation via Matlab is done first of all dividing the path $\underline{x}(t)$ in in steps $\underline{x}(t_i)$ with $i \in \{0, 1, \dots, n\}$, then we set the beginning point $\underline{y}(t_0)$, and then to find the point $\underline{y}(t_{i+1})$ we look for the intersection between the circumference with center $\underline{x}(t_{i+1})$ and radius l , and the line passing trough $\underline{x}(t_i)$ and $\underline{y}(t_i)$. We will find two points, but we choose the one close to $\underline{y}(t_i)$.

In order to decide the geometry of the roundabout (size), we need to describe the trajectories of the vehicle that will determine the minimum size of the circle.

First we try to describe a Point moving in the circle. To approach the point, we wrote a **Matlab Program** which gives a point approaching and moving around a circle.

The next step is trying to figure out how the rear (back) wheels will follow the front ones. The idea is based on the assumption that for every small steps, the trajectory can be approximated in the following ways or using the following assumptions:

- (a) the speed of vehicles used in the model is a constant;
- (b) the car is led by the front wheels;
- (c) at each instant the direction of the back wheels is decided as explained above.

After executing the Matlab programs, we get a good result: the car (suggested by a vector) seems to move properly. On top of it, we built a car over the axes (in a rectangular form) so that we can see how much area it covers. The same program also gives a graph showing how the curvature of the wheels depends on time.

Till now the car is moving in general curves: the next step is to describe how the trajectory enters into the roundabout.

The last step is to decide the curve linking the straight line with the circle. We first considered circles, but we realized that this model was trivial because it would correspond to an instant stirring of the wheels (which is not possible). Thus we decided to try with polynomials. In this case we had the possibility to give some boundary conditions about the directions and curvature, resulting in a 5th degree polynomial. That seems to be a good result, but the program performing the simulation is not flexible enough as the parameters change. This led us to consider another approach, i.e. a program that at a certain point it decides to start to stir and the curvature increases smoothly, then after some point it starts to decrease until it's a constant.

Points where the direction and curvature will change are depending on the size of the roundabout and position of the car.

The problem is now just to give the suitable values for the parameters x and y . Here we have a look at the numerical outputs(Figure2 and 3) from our matlab codes:

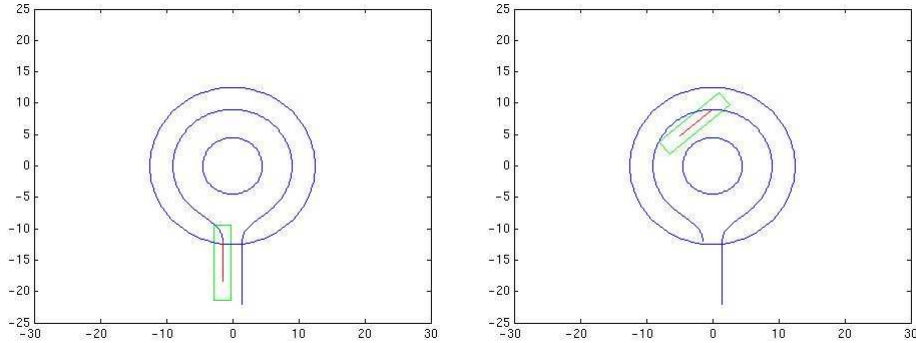


Figure 2: A car(bus) entering and moving in the roundabout

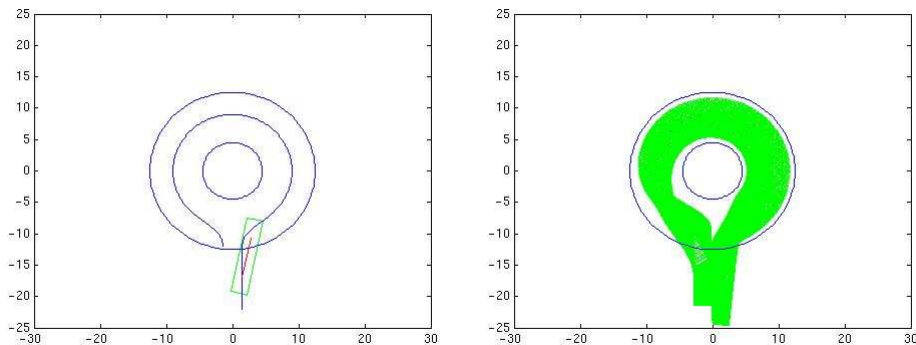


Figure 3: A car(bus) moving out of the roundabout and the covering area the bus.

Figures 2,3 shows a bus passing through a roundabout (sizes of the bus and minimum stirring radius are taken from the website of the Italian company: "www.bredamenarinibus.it"). From the Figure 3, we can see that we can decide the shape and size of roundabouts if we know the covering area of the cars.

3.2 Method 2

Now we will describe the trajectories of the vehicle in a different way. Firstly let's introduce some notation. In Figure 4, any part of the path of the bus around the roundabout we use the following notations: P and Q represent the front wheel and back wheel respectively. V is the speed of the bus in the neighborhood of this front wheel. v is the speed of the front wheels; θ is the angle between the length of the vehicle and the direction of the front wheel, $v_1 = v\cos\theta$ is the speed along the length of the vehicle, $v_2 = v\sin\theta$ is the speed perpendicular with the length of the vehicle; \hat{n} is the unit vector perpendicular with the length of the vehicle, and \hat{m} is the unit vector parallel to the length of the vehicle.

We use an iterative algorithm to calculate the positions of the front wheel and rear wheel for each time step Δt as the following:

$$\begin{aligned}
 Px_i &= Px_{i-1} + v_1 \Delta t m_x + v_2 \Delta t n_x; \\
 Py_i &= Py_{i-1} + v_1 \Delta t m_y + v_2 \Delta t n_y; \\
 Qx_i &= Qx_{i-1} + v_1 \Delta t m_x; \\
 Qy_i &= Qy_{i-1} + v_1 \Delta t m_y,
 \end{aligned}$$

in which (Px_i, Py_i) and (Qx_i, Qy_i) are the coordinates of the front wheel and the rear wheel at $i\Delta t$ time step, respectively.

We can determine the moving path of the vehicle by changing the angle θ , which is very similar with the reality (drivers control the car by steering wheels). After knowing how the vehicle move, we can find a minimal size (we should guarantee that the vehicle can enter into and get out of the roundabout) of the roundabout for a particular vehicle, since every kind of vehicle has a minimal turning radius. For different vehicles we can change the length, consequently we can get a minimal size for all kinds of vehicles. By this model we can also find how the speed of the vehicle influences the size of the roundabout: the size should be increased as the speed increases. This is also consistent with the reality, because it is hard for the driver to change the front wheel to a large angle in a short time if the speed is too high. To decrease the speed of vehicles is a main purpose of using the roundabouts, so we should consider the speed factor.

Here we have a look at the output(Figure4,5 and 6) from our matlab codes (the red line represents the vehicle and the blue curves represent the path of the front and the back wheels, respectively):

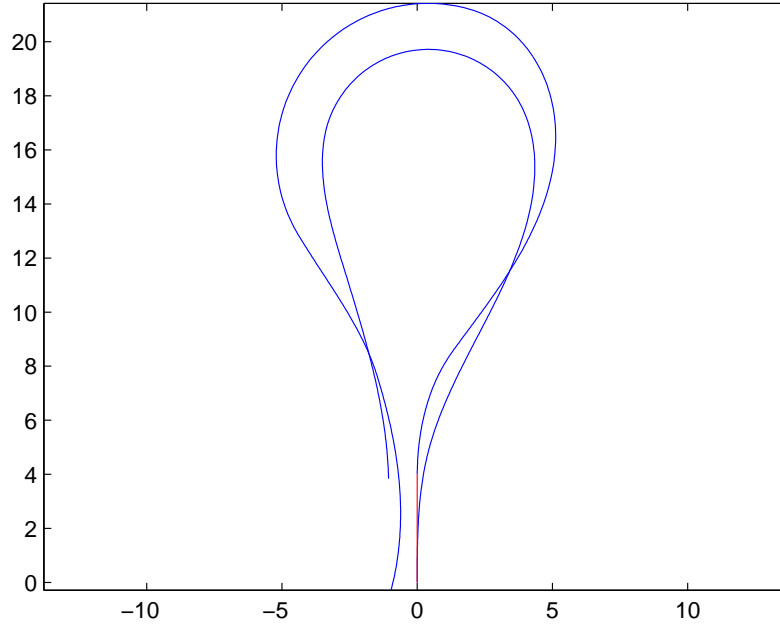


Figure 4: The car is entering into a roundabout

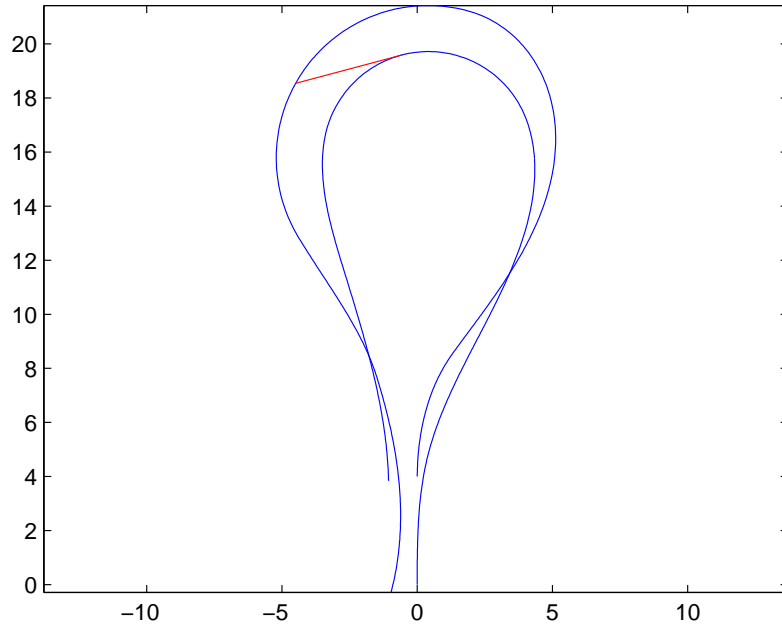


Figure 5: The car is moving in a roundabout

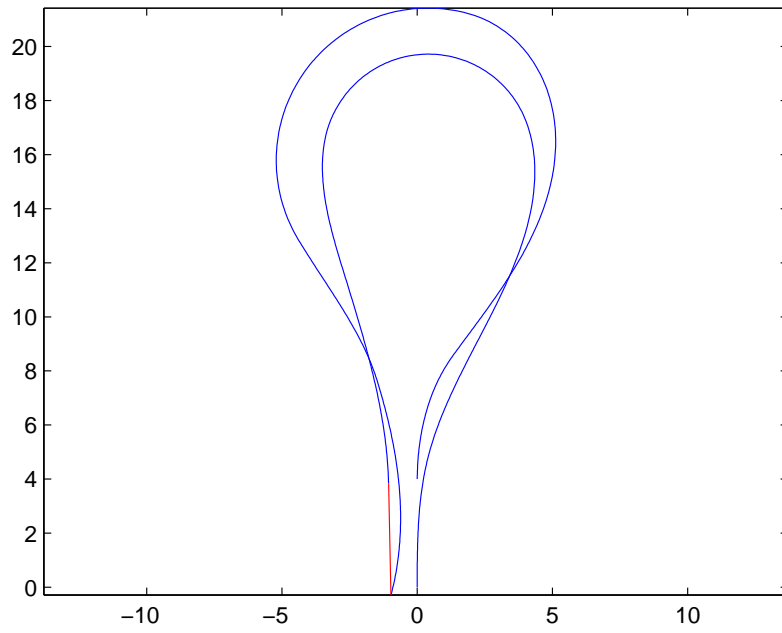


Figure 6: The car is going out of a roundabout

3.3 Comparison of the two Models

Looking at the two models above, we observe that each of these two models has got a number of merits and demerits. We will give a few of them:

For the **first method**, the advantages for this model is that, if the path of the front wheels is given, then the one for rear wheels is determined automatically for the given size of the car (even so for different sizes, we have different pathes).

The disadvantages of this first model are that we have to choose a good initial position for the rear wheels $y(t_0)$, so that the length of the car is correct, and there are no strange relation with the path $\underline{x}(\bar{t})$ (it is not possible that the front wheels are turning in a perpendicular way relative with the position of the car). Also we have to pay attention to the path $\underline{x}(t)$ its self. This has to be a smooth path, with a continuous second derivative, otherwise the car would be able to outright change the turning circle, situation not possible in the real world.

For the **second method**, the advantage is that we can control the paths of vehicles freely by changing the angle (θ) between the length of the vehicle and the direction of the front wheels (like drivers control the direction of the cars by steering the wheels), so we can move the vehicle to anywhere we need theoretically. However we need to assume that the vehicle has no acceleration.

4 Future study

We can update our models by considering the following special cases:

1. how the velocity will affect the radius of the circular path (and hence maximum velocity depending on the size of the roundabout);
2. the available natural and geographical factors, e.g. buildings, trees, mountains, etc
3. cars with more wheels (compared to the bus), like 18 wheeled trucks.

5 Appendix: Matlab codes

```
method 1: function esperimento nf=600;
t=linspace(-10,10,nf);
lcar=6.32;
fb=2.560;
rb=3.08;
wil=2.5/2;
[fx,fy]=curva(nf/3);
fig1=figure(1);
winsize = get(fig1,'Position');
winsize(1:2) = [0 0];
A=moviein(nf);
fig1=figure(1);
winsize = get(fig1,'Position');
winsize(1:2) = [0 0];
A=moviein(nf);
bx=zeros(1,nf);
by=bx;
dir=[fx(2)-fx(1) fy(2)-fy(1)];
dir=dir/norm(dir);
bx(1)=fx(1)-dir(1)*lcar;
by(1)=fy(1)-dir(2)*lcar;
for i=2:nf
h=fminsearch(@(cin) abs(norm([fx(i) fy(i)]-([bx(i-1) by(i-1)]+cin*dir))-lcar),0);
bx(i)=bx(i-1)+dir(1)*h;
by(i)=by(i-1)+dir(2)*h;
dir=[fx(i)-bx(i) fy(i)-by(i)];
dir=dir/norm(dir);
```

```

end Azzo=zeros(nf,2); B=zeros(nf,2); C=zeros(nf,2); D=zeros(nf,2); for i=1:nf

dir=[fx(i)-bx(i) fy(i)-by(i)];

dir=dir/norm(dir); n=[-dir(2) dir(1)];

Azzo(i,:)= [bx(i)-dir(1)*rb-n(1)*wil by(i)-dir(2)*rb-n(2)*wil]; B(i,:)= [fx(i)+dir(1)*fb-n(1)*wil fy(i)+dir(2)*fb-
n(2)*wil]; C(i,:)= [fx(i)+dir(1)*fb+n(1)*wil fy(i)+dir(2)*fb+n(2)*wil]; D(i,:)= [bx(i)-dir(1)*rb+n(1)*wil
by(i)-dir(2)*rb+n(2)*wil]; plot([Azzo(i,1) B(i,1) C(i,1) D(i,1) Azzo(i,1)], [Azzo(i,2) B(i,2) C(i,2)
D(i,2) Azzo(i,2)], 'g') hold on plot([25 -25], [25 -25], '*w') hold on plot(fx,fy) hold on plot([fx(i)
bx(i)], [fy(i) by(i)], 'r') hold on

A(:,i)=getframe(fig1,winsize);

hold off end movie(fig1,A,nf,5,winsize)

tanv=zeros(nf-1,2); acc=zeros(nf-2,2); for i=1:nf-1 tanv(i,:)= [fx(i+1)-fx(i) fy(i+1)-fy(i)]; end for
i=1:nf-2 acc(i,:)=tanv(i+1,:)-tanv(i,:); end cur=zeros(1,nf-2); for i=1:nf-2 X=cross([fx(i+2)-fx(i+1)
fy(i+2)-fy(i+1) 0], [fx(i+1)-fx(i) fy(i+1)-fy(i) 0]); cur(i)=sign(X(3))*norm(X)/norm([fx(i+1)-fx(i)
fy(i+1)-fy(i)]); end vel=zeros(1,nf-1); for i=1:nf-1 vel(i)=norm(tanv(i,:)); end

function esperimento

nf=600;

t=linspace(-10,10,nf);

lcar=6.32;

fb=2.560;

rb=3.08;

wil=2.5/2;

[fx,fy]=curva(nf/3);

fig1=figure(1);

winsize = get(fig1,'Position');

winsize(1:2) = [0 0];

A=moviein(nf);

fig1=figure(1);

winsize = get(fig1,'Position');

winsize(1:2) = [0 0];

A=moviein(nf);

bx=zeros(1,nf);

by=bx;

```

```

dir=[fx(2)-fx(1) fy(2)-fy(1)];
dir=dir/norm(dir);
bx(1)=fx(1)-dir(1)*lcar;
by(1)=fy(1)-dir(2)*lcar;
plot([25 -25],[25 -25],'*w') hold on plot(fx,fy) hold on
for i=2:nf
h=fminsearch(@(cin) abs(norm([fx(i) fy(i)]-[bx(i-1) by(i-1)]+cin*dir))-lcar),0);
bx(i)=bx(i-1)+dir(1)*h;
by(i)=by(i-1)+dir(2)*h;
dir=[fx(i)-bx(i) fy(i)-by(i)];
dir=dir/norm(dir);
end Azzo=zeros(nf,2); B=zeros(nf,2); C=zeros(nf,2); D=zeros(nf,2); for i=1:nf
dir=[fx(i)-bx(i) fy(i)-by(i)];
dir=dir/norm(dir); n=[-dir(2) dir(1)];
Azzo(i,:)= [bx(i)-dir(1)*rb-n(1)*wil by(i)-dir(2)*rb-n(2)*wil]; B(i,:)= [fx(i)+dir(1)*fb-n(1)*wil fy(i)+dir(2)*fb-
n(2)*wil]; C(i,:)= [fx(i)+dir(1)*fb+n(1)*wil fy(i)+dir(2)*fb+n(2)*wil]; D(i,:)= [bx(i)-dir(1)*rb+n(1)*wil
by(i)-dir(2)*rb+n(2)*wil]; plot([Azzo(i,1) B(i,1) C(i,1) D(i,1) Azzo(i,1)], [Azzo(i,2) B(i,2) C(i,2)
D(i,2) Azzo(i,2)], 'g') hold on
A(:,i)=getframe(fig1,winsize);
end movie(fig1,A,nf,5,winsize)
tanv=zeros(nf-1,2); acc=zeros(nf-2,2); for i=1:nf-1 tanv(i,:)= [fx(i+1)-fx(i) fy(i+1)-fy(i)]; end for
i=1:nf-2 acc(i,:)=tanv(i+1,:)-tanv(i,:); end cur=zeros(1,nf-2); for i=1:nf-2 X=cross([fx(i+2)-fx(i+1)
fy(i+2)-fy(i+1) 0],[fx(i+1)-fx(i) fy(i+1)-fy(i) 0]); cur(i)=sign(X(3))*norm(X)/norm([fx(i+1)-fx(i)
fy(i+1)-fy(i)]); end vel=zeros(1,nf-1); for i=1:nf-1 vel(i)=norm(tanv(i,:)); end
Method 2: clear all; nd=50; nf=150; ng=135; nh=60; ni=80; nn=0;
v=1;
tt=0.1;
fig1=figure(1);
winsize = get(fig1,'Position');
winsize(1:2) = [0 0];
A=moviein(nd+nf+ng+nh);

```

```

fx=zeros(1,nd+nf+ng+nh);
fy=fx;
bx=zeros(1,nd+nf+ng+nh);
by=bx;
fx(1)=0; fy(1)=4;
bx(1)=0; by(1)=0;
m=[fx(1)-bx(1) fy(1)-by(1)];
m=m/norm(m);
n=[fy(1)-by(1) -(fx(1)-bx(1)) ];
n=n/norm(n);
for i=2:nd
v1=v*cos(pi*i/(8*nd)); v2=v*sin(pi*i/(8*nd));
fx(i)=fx(i-1)+v1*tt*m(1)+v2*tt*n(1);
fy(i)=fy(i-1)+v1*tt*m(2)+v2*tt*n(2);
bx(i)=bx(i-1)+v1*tt*m(1);
by(i)=by(i-1)+v1*tt*m(2);
m=[fx(i)-bx(i) fy(i)-by(i)];
m=m/norm(m);
n=[fy(i)-by(i) -(fx(i)-bx(i)) ];
n=n/norm(n); end
for i=nd+1:nd+nf
v1=v*cos(pi/8-pi/8*(i-nd)/nd); v2=v*sin(pi/8-pi/8*(i-nd)/nd);
fx(i)=fx(i-1)+v1*tt*m(1)+v2*tt*n(1);
fy(i)=fy(i-1)+v1*tt*m(2)+v2*tt*n(2);
bx(i)=bx(i-1)+v1*tt*m(1);
by(i)=by(i-1)+v1*tt*m(2);
m=[fx(i)-bx(i) fy(i)-by(i)];
m=m/norm(m);

```

```

n=[fy(i)-by(i) -(fx(i)-bx(i)) ];
n=n/norm(n);
end
for i=nd+nf+1:nd+nf+ng phi=pi/8-pi/8*nf/(nd-1); v1=v*cos(phi); v2=v*sin(phi);
fx(i)=fx(i-1)+v1*tt*m(1)+v2*tt*n(1);
fy(i)=fy(i-1)+v1*tt*m(2)+v2*tt*n(2);
bx(i)=bx(i-1)+v1*tt*m(1);
by(i)=by(i-1)+v1*tt*m(2);
m=[fx(i)-bx(i) fy(i)-by(i)];
m=m/norm(m);
n=[fy(i)-by(i) -(fx(i)-bx(i)) ];
n=n/norm(n);
end
for i=nd+nf+ng+1:nd+nf+ng+nh
v1=v*cos(phi-(i-nf-ng-nd)*phi/nh); v2=v*sin(phi-(i-nf-ng-nd)*phi/nh);
fx(i)=fx(i-1)+v1*tt*m(1)+v2*tt*n(1);
fy(i)=fy(i-1)+v1*tt*m(2)+v2*tt*n(2);
bx(i)=bx(i-1)+v1*tt*m(1);
by(i)=by(i-1)+v1*tt*m(2);
m=[fx(i)-bx(i) fy(i)-by(i)];
m=m/norm(m);
n=[fy(i)-by(i) -(fx(i)-bx(i)) ]; n=n/norm(n);
end for i=nd+nf+ng+nh+1:nd+nf+ng+nh+ni v1=v*cos((i-nf-ng-nh-nd)*pi/(11*ni)); v2=v*sin((i-
nf-ng-nh-nd)*pi/(11*ni));
fx(i)=fx(i-1)+v1*tt*m(1)+v2*tt*n(1);
fy(i)=fy(i-1)+v1*tt*m(2)+v2*tt*n(2);
bx(i)=bx(i-1)+v1*tt*m(1);
by(i)=by(i-1)+v1*tt*m(2);
m=[fx(i)-bx(i) fy(i)-by(i)];

```

```
m=m/norm(m);  
n=[fy(i)-by(i) -(fx(i)-bx(i)) ];  
n=n/norm(n); end for i=1:nd+nf+ng+nh+ni  
plot(fx,fy) axis equal hold on plot(bx,by) hold on plot([fx(i) bx(i)],[fy(i) by(i)],'r','EraseMode',  
'xor') axis equal hold on A(:,i)=getframe(fig1,winsize); hold off end  
movie(A)
```

References

- [1] JOHN BAYLIS: The mathematics of a driving hazard, *Math. Gaz.*, 57 (1973), pp. 23 - 26.
- [2] EDWARD A. BENDER: A driving hazard revisited, *SIAM Rev.*, 54 (1979), pp. 136 - 138.
- [3] H. I. FREEDMAN AND S.D. RIEMENSCHNEIDER: Determining the path of the rear wheels of a bus, *SIAM Rev.*, 25 (1983), pp. 561-568.
- [4] S. CHOWLA UND H. J. RYSER: Combinatorial Problems, *Can. J. Math.*, 2 (1950), 93 - 99
- [5] HENRY B. MANN: Necessary Conditions for the Existence of Difference Sets, *Addition Theorems* (1965), 72 - 86