# Roundabout Efficiency

# New Mexico
# Supercomputing Challenge
# Final Report
# March 25, 2012

# Team 14
# 5th Grade
# Aspen Elementary School, Los Alamos, NM

**Team Members:**
Christopher Koh
Do Vo
Jesse Prime

**Sponsoring Teachers:**
Mrs. Thomas
Mrs. Vandenkieboom

**Project Mentors:**
Mike Prime
Duc Vo

# Executive Summary:

Saving time is saving money. Our team thinks roundabouts might be more efficient than stoplights. If so, Los Alamos County could add roundabouts to intersections on Trinity Drive, and we'd like to figure out if this is a good idea.

To simulate this project, we made two simulations using NetLogo®. One simulation has separate right turn lanes and one doesn't because we thought the first simulation had slightly overestimated capacity. This roundabout is a one lane roundabout with one lane of traffic going in each direction. We made three different cars red (right), green (straight), and blue (left).

To test the roundabout capacity, we varied the amount of traffic (number of cars per mile) but the color and placement is random. As soon as the simulation ends at 30,001 ticks (one tick = 1/100 seconds), we collect the number of cars per minute and compare it to the number of cars on Trinity Drive.

Our simulation shows a total capacity of 60-80 cars per minute for four directions of traffic flow in the roundabout. This is close to peak traffic flows measured on Trinity drive. Based on these results, a roundabout might handle the traffic flow on Trinity Drive. We think it is worth further study. If we did further study we would improve the model.

# Statement of the problem:

Most people don't think of roundabouts instead of stoplights. A better intersection means less time waiting. Our team thinks that in some cases roundabouts can work better than stoplights.

Our idea for roundabout efficiency came from the Los Alamos newspaper when Los Alamos County was thinking about putting roundabouts on Trinity Dr. Our goal in our project was to find the number of cars that can go through a roundabout in a given time without bad traffic flow. Our team studied a four-way roundabout with one lane of traffic going in each direction with normal driving rules. Our data for the project was based on the traffic flow on Trinity Dr.

We compared our model to the traffic flow on Trinity Dr. Figure 1 shows traffic counts taken on Thursday, August 12, 2010 by The Los Alamos Dept. of Public Works. This shows the number of cars per hour going eastbound on Trinity Dr. near Ashley Pond. The peak traffic flow of about 1300 cars per hour occurs at lunch time.

*Figure 1. Traffic flow data eastbound on Trinity Dr.*

# Method to Solve:

We created a simulation of a four-way roundabout. First, we tried to use StarLogo TNG®, but then we switched to NetLogo® because it was easier. We looked up and programmed the realistic details and features of a roundabout that includes ten foot wide lanes. The ideal roundabout lane is ten feet wide because speed is lower than on a twelve foot highway road and ten is an easy number to work with. Then we programmed the cars to follow the rules of the road. We calculated the number of cars that went through our roundabout per minute.

Figure 2 shows our model roundabout with right turn lanes.  We randomly distributed cars in appropriate lanes. Red cars turn right in the roundabout, green cars go straight, and blue cars turn left.

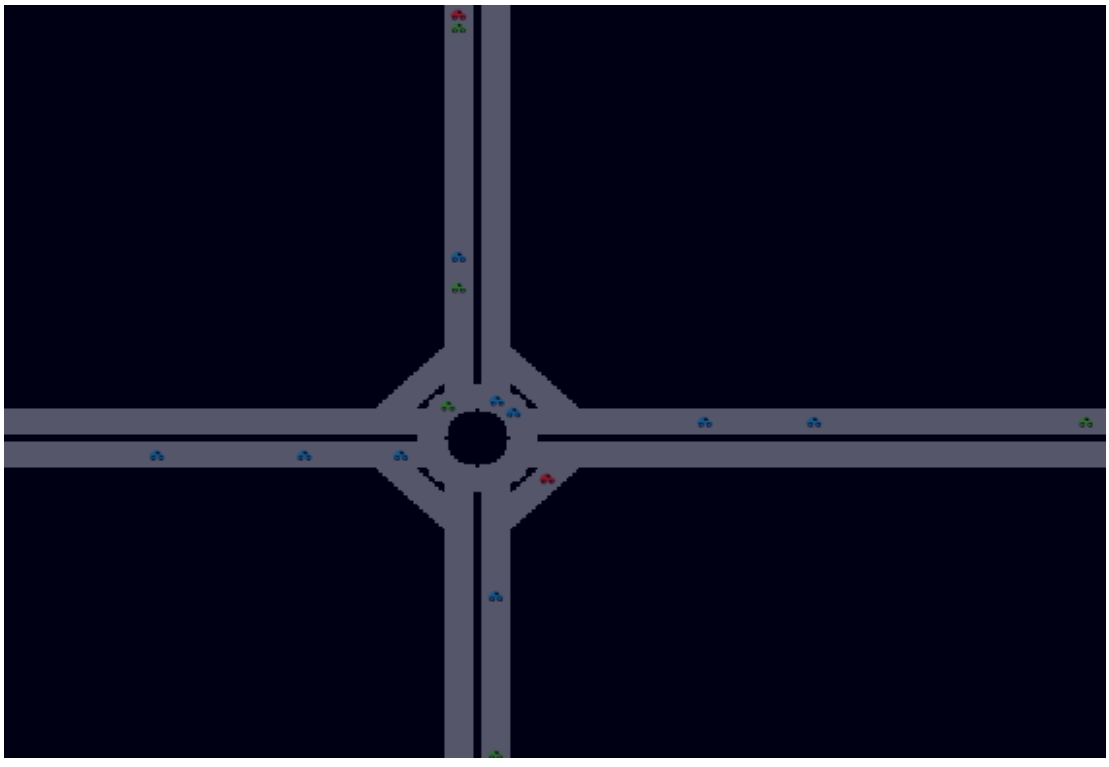

*Figure 2. NetLogo roundabout with separate right turn lanes (zoomed in).*

Sometimes there is not enough room for a right turn lane so people turning right

have to go inside the roundabout. Figure 3 shows our simulation of a roundabout without right turn lanes. We will compare the two models to see how much additional capacity we get by adding right turn lanes.

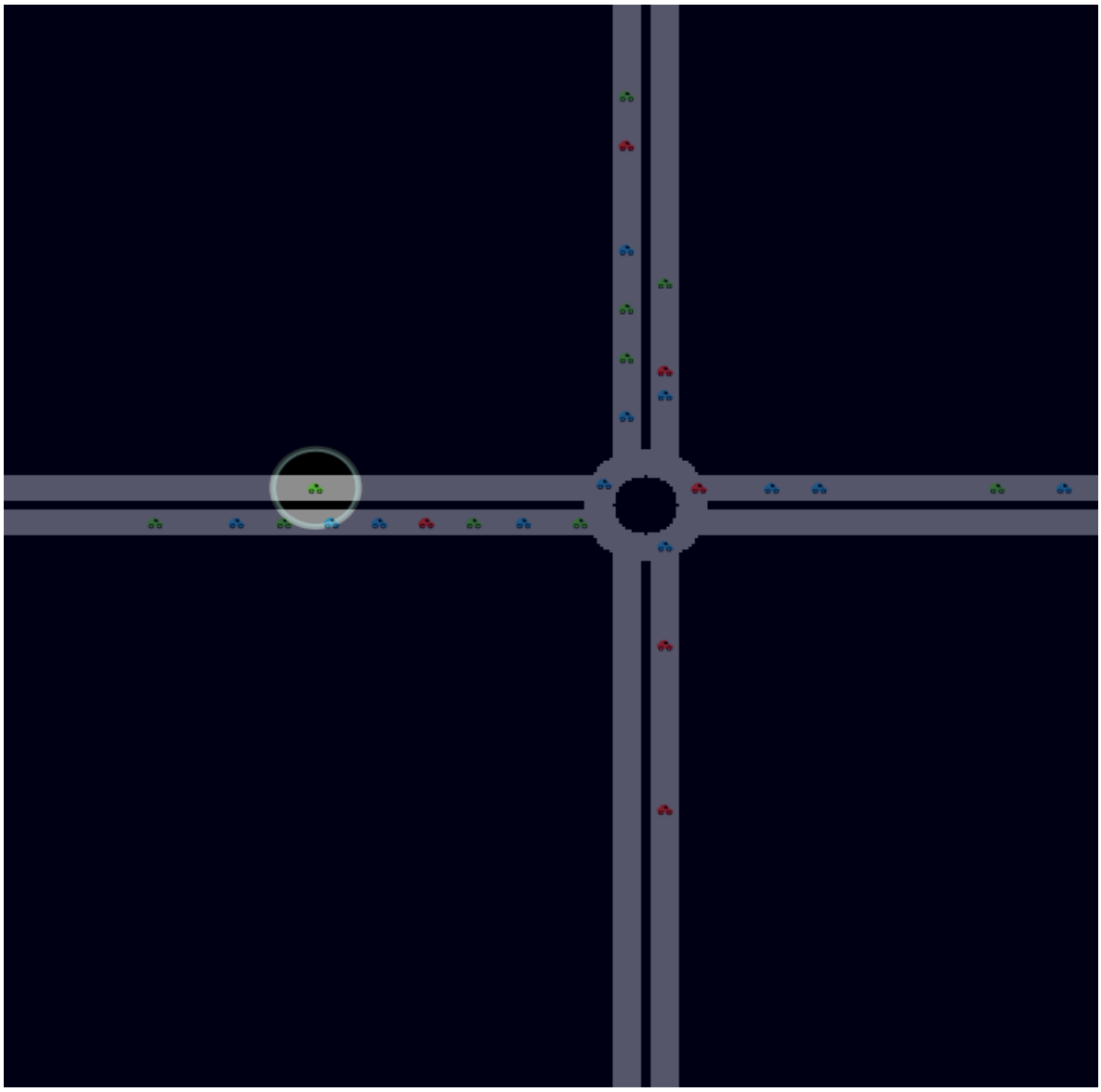

*Figure 3. NetLogo roundabout without right turn lanes (zoomed in).*

Now that the model is initialized, the simulation follows the flow chart in Figure 4. For all driver behavior, we allowed one second reaction time. The maximum speed of the cars is 35 mile/hour on a straight road and half of that speed in the roundabout. Cars

will slow down when another car is closer than one car length ahead plus the distance covered in one second. If the car in front of them or roundabout is too close, the car will slow down or stop. The typical deceleration is ten feet per second per second (at this deceleration, a car traveling at 35 mph will take 5 seconds to come to a complete stop). In some situations the cars may have to slow down faster. When a car is approaching the roundabout it will decelerate using this equation:

$$a = \frac{v_f^2 - v^2}{2d}$$

Where $a$ is the deceleration, $v_f$ is the final velocity which in this case is the maximum speed in the roundabout, $v$ is the current velocity, and $d$ is the distance to the roundabout. When they reach the roundabout they will check for cars coming from the left. Based on the distance and speed of such cars, they will enter the roundabout if it's safe or else stop and then resume when it is safe. Within the roundabout, the cars maintain a minimum separation of the length of a car. When exiting the roundabout, the cars enter the main road if it's safe, if not, the car waits for a chance to go. When on the main road, the car accelerates to maximum speed if there are no cars in its way.

We started our project in October 2011 and we have spent roughly 45 hours programming and 25 on research and documentation.

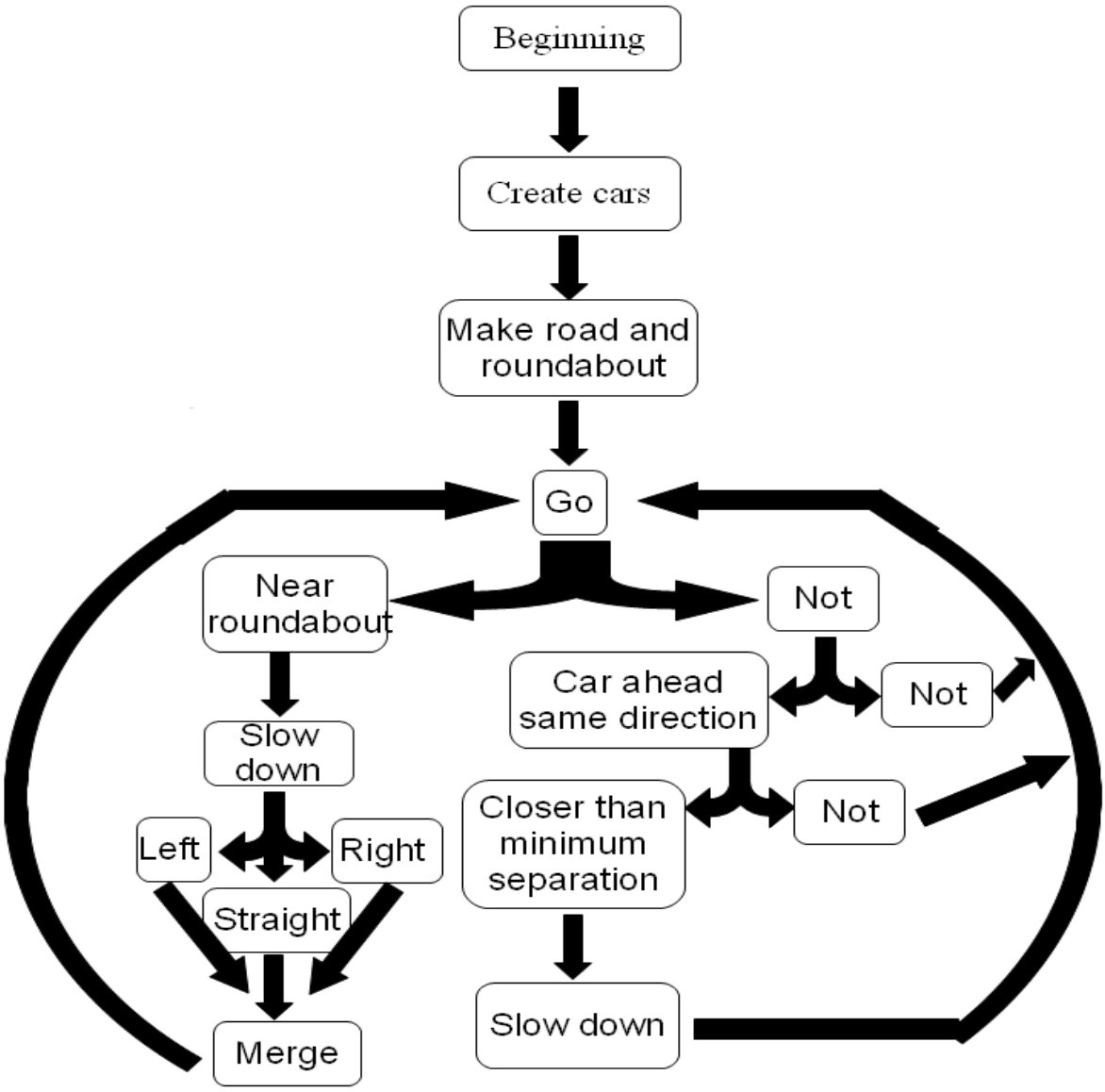


*Figure 4. Flow chart on how simulation works.*

# Results:

In order to estimate the traffic flow capacity with our model, we varied the initial

number of cars in the simulation. As we increased the number of cars, the traffic started to back up. In the program, we counted the cars that went through the roundabout in five simulated minutes. From there, we obtained the traffic flow in cars per minute. Because the result depended on the random distribution of the cars, we ran each test three times.

Figure 5 shows the results for our simulations of a roundabout with right turn lanes (see figure 2). When the number of cars in the simulation exceeds about 40 cars per mile, the traffic gets worse, and the results depend on the initial conditions. We think that our simulated roundabout with right turn lanes can handle about 80 cars per minute. However, some cars overlapped each other when cars making a right turn merged back into the main road. We were unable to control this problem. Because of this, we think our results overestimate the capacity slightly.

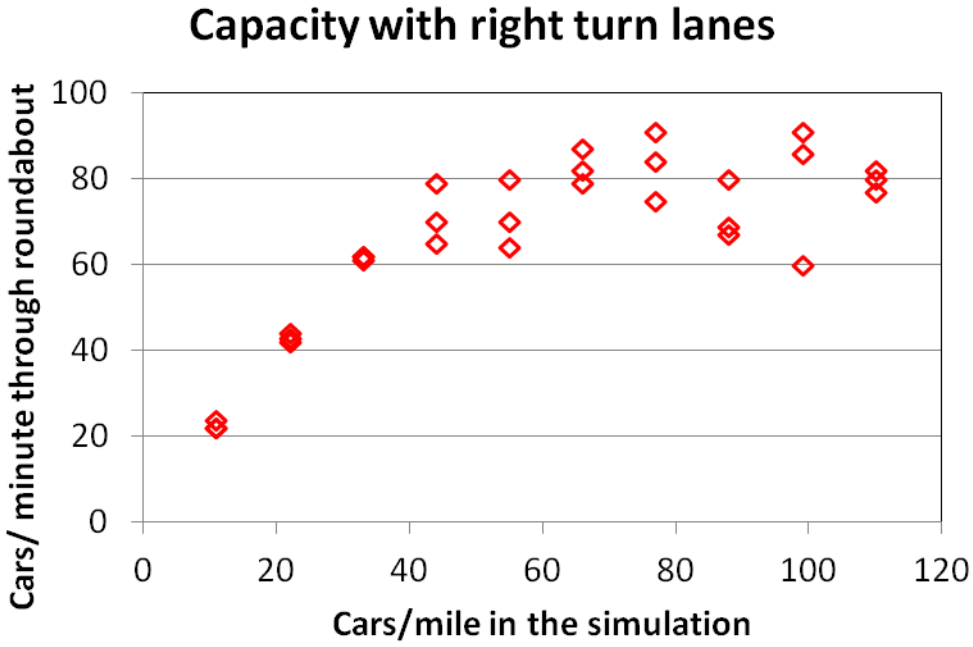


*Figure 5. Our simulated traffic flow for a four-way roundabout with right turn lanes.*

In order to obtain a more conservative estimate of traffic flow, we ran the simulation without right turn lanes. Figure 6 shows the results without right turn lanes.

Now the traffic capacity is about 60 cars per minute. Since the roundabout design on Trinity has right turn lanes, this is probably underestimated.

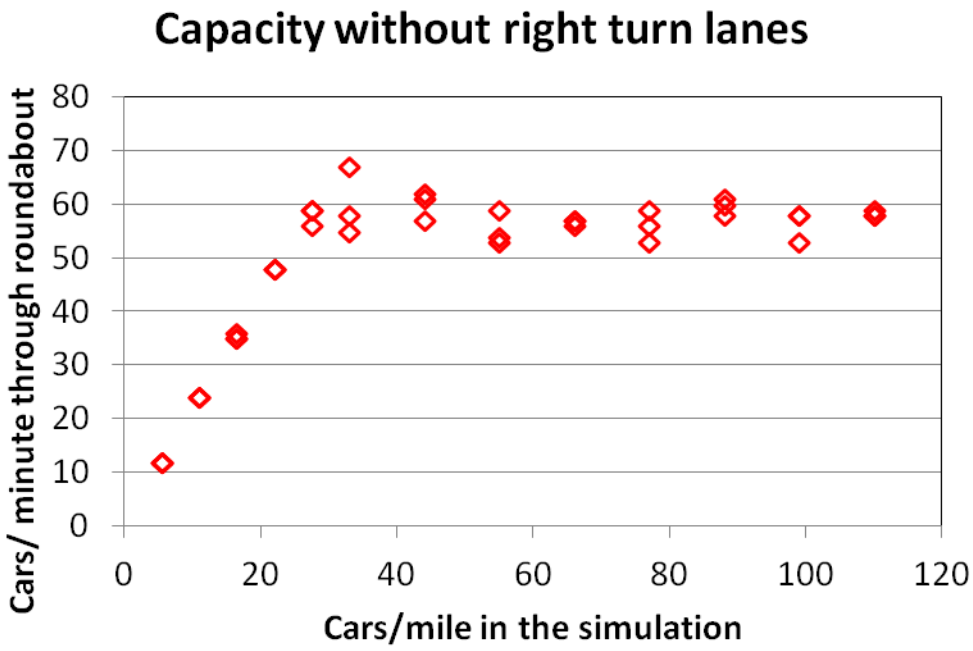


*Figure 6. Our simulated traffic flow for a four-way roundabout without right turn lanes.*

# Conclusions:

Our simulation shows a total capacity of 60-80 cars per minute for four directions of traffic flow in the roundabout. The data on Trinity Drive shows a peak of 1300 cars per hour, or about 22 cars per minute, eastbound at lunch time. Based on our own observations, the westbound traffic can be equal to eastbound traffic. North-south traffic at the intersections is quite small. Therefore, we conservatively multiply by 3 to arrive at 66 cars per minute coming from all directions. Based on these results, a roundabout might handle the traffic flow on Trinity Drive. We think it is worth further study.

If we did further study we would improve the model. We would add pedestrians, cops, speeders, bicyclist, and emergency vehicles. We would consider accidents. We would also simulate uneven traffic from different directions.

## Most significant original achievement:

Our team's greatest achievement is making the cars correctly know when it is safe to go into the roundabout. To do this we first calculated, for each car approaching the roundabout, how much time until the car would enter the roundabout. Then we checked all the cars in the roundabout to see if any cars would be within one car length at that time. A car would only enter the roundabout if there was a one car length margin.


## Acknowledgments

We would like to give our thanks to:

**Sponsoring Teachers:**

Mrs. Thomas

Mrs. Vandenkieboom

**Project Mentor:**

Mike Prime

Duc Vo

## Editor:

Tracy Koh

# Bibliography:


Los Alamos County, and New Mexico Planning And Engineering. "Comprehensive Transportation Study and Plan for NM502." *Comprehensive Transportation Study and Plan for NM502*. 16 Nov. 2010. pages 7,21,30,42. Web. 4 Dec. 2011. <http://www.losalamosnm.us/projects/publicworks/Documents/Presentation_Final111610.pdf>.

Los Alamos County. "NM502/Trinity Drive Corridor Study (includes the East Road Sound Mitigation Study as Well as the Central/Oppenheimer Improvement Study)." *NM502/Trinity Drive Study as Well as the Corridor Study (includes the East Road Sound Mitigation Central/Oppenheimer Improvement Study)*. regularly edited. Web. 04 Dec. 2011. <http://www.losalamosnm.us/projects/publicworks/Pages/NM502TrinityDriveCorridorStudy.aspx>.

McKenna, Arin. "Ourston Assesses Roundabout Potential." Los Alamos Monitor [Los Alamos] 1 Dec. 2011, A1 sec.: A1 . Print. http://www.lamonitor.com/content/ourston-assesses-roundabout-potential

NetLogo Home Page. Wilensky, U. (1999). NetLogo. http://ccl.northwestern.edu/netlogo/. Center for Connected Learning and Computer-Based Modeling, Northwestern University, Evanston, IL.

New Mexico Department of Transportation. "New Mexico Department of Transportation - Driving in Roundabouts." *New Mexico Department of Transportation - Home Page*. regularly edited. Web. 04 Dec. 2011. <http://nmshtd.state.nm.us/main.asp?secid=15792>.

Data from Los Alamos county on Trinity Drive traffic: http://wcmead.org/nm502/nm502-capacity3.htm

# Appendices:

## Code for Roundabout with right turn lanes

```
 globals [ gl_my-car gl_cars-count gl_accelerator gl_decelerator gl_speed-max gl_speed-min gl_speed-
maxcircle gl_speed-maxright gl_distance-straight
  gl_distance-left gl_distance-right gl_degree-per-foot gl_side gl_slowdown gl_straightdistance
gl_distanceto00 gl_minseparation gl_reactiontime]
breed [cars car]
turtles-own [ speed droundabout headingdirection ]

to setup
  clear-all
  set gl_cars-count 0
  ; 100 ticks per second
  set gl_accelerator 0.001 ; 10 ft per second per second
  set gl_decelerator 0.001 ; 10 ft per second per second
  set gl_speed-max  0.5133333333333 ; 35mph, 51.3333 ft per second
  set gl_speed-min  1.e-10
  set gl_speed-maxcircle  gl_speed-max * 0.5  ; half normal speed
  set gl_speed-maxright  gl_speed-max * 0.62  ; about 60% normal speed
  set gl_distance-straight 34.77
  set gl_distance-left 58.34
  set gl_distance-right 35.2
  set gl_degree-per-foot 3.81972
  set gl_side sqrt (15 ^ 2 - 6 ^ 2)
  set gl_slowdown (gl_speed-max ^ 2 - gl_speed-maxcircle ^ 2) / (2 * (gl_decelerator))
  set gl_straightdistance gl_slowdown + gl_side
  set gl_distanceto00 sqrt (gl_straightdistance ^ 2 + 6 ^ 2)
  set gl_minseparation 15; assume car lenght 10' & distance between cars 5'
  set gl_reactiontime 100; 100 ticks or 1 second reaction time

  create-cars NumberCars
  [
    set color (random 3 * 40 + 15) ; 15=red=go right, 55=green=go straight, 95=sky=go left
    set shape "car"
    set size 5
    set speed gl_speed-max - random-float .1
    set droundabout 0
    set headingdirection 0
    distribute-cars
  ]
  set gl_my-car one-of cars
  watch gl_my-car
  ask patches
```

```
  [
    if (pxcor ^ 2 + pycor ^ 2 < 20 ^ 2) and (pxcor ^ 2 + pycor ^ 2 > 10 ^ 2)
    [
    set pcolor grey
    ]
    if abs (pxcor) > 1 and abs (pxcor) < 11 and (pxcor ^ 2 + pycor ^ 2 > 10 ^ 2)
    [
      set pcolor grey
    ]
    if abs (pycor) > 1 and abs (pycor) < 11 and (pxcor ^ 2 + pycor ^ 2 > 10 ^ 2)
    [
      set pcolor grey
    ]
    if ((abs (pxcor) - 44 < pycor) and (abs (pxcor) - 30 > pycor) and abs (pxcor) > 1 and pycor < -1)
    [
      set pcolor grey
    ]
    if (( (- pxcor) + 44 > pycor) and ( (- pxcor) + 30 < pycor) and (pxcor) > 1 and pycor > 1)
    [
      set pcolor grey
    ]
    if (( ( pxcor) + 44 > pycor) and ( ( pxcor) + 30 < pycor) and  (pxcor) < -1 and pycor > 1)
    [
      set pcolor grey
    ]
  ]
end

to distribute-cars  ;; procedure
    set heading random 4 * 90
    if (heading = 0)
    [setxy 6 (38 + random (max-pycor - 38)) * (2 * random 2 - 1)]
    if (heading = 90)
    [setxy ((38 + random (max-pxcor - 38)) * (2 * random 2 - 1)) -6]
    if (heading = 180)
    [setxy -6 ((38 + random (max-pycor - 38)) * (2 * random 2 - 1))]
    if (heading = 270)
    [setxy ((38 + random (max-pxcor - 38)) * (2 * random 2 - 1)) 6]
    if any? other turtles-here
    [distribute-cars ]
end

to go
  ask cars [goall]
  tick
  if ticks > 30000 [stop]
end
```

```
to goall
  let myhd heading
  set headingdirection (subtract-headings myhd (towardsxy 0 0))
  ifelse color = red and distancexy 0 0 < 31.7
  [
    goright
  ]
  [
    ifelse distancexy 0 0 <= 15
    [
      gostraightleft
    ]
    [
      gomainroad
    ]
  ]
  plot [speed] of gl_my-car * 360000 /  5280 ; convert from feet per tick to mph
end

to goright
;  if [ pcolor ] of patch-ahead 23 != grey
  if headingdirection > 0 and headingdirection < 45 ; begin right turn
  [
    set gl_cars-count gl_cars-count + 1
    set droundabout 0
    rt 45
  ]
  ifelse droundabout < gl_distance-right
  [
    let myhd heading
    let d gl_distance-right - droundabout
    ifelse d > 12
    [
      let t d / speed
      let cars-45degrees cars with [(subtract-headings heading myhd = 45) and (distance myself < 52.35)]
      let cars-ininterval cars-45degrees with [(distancexy 0 0) + speed * t > 15 and (distancexy 0 0)  +
speed * t < 46.39]
      ifelse any? cars-ininterval
      [
        let deceleration-to-stop (speed ^ 2 / (2 * (d - 12)))
        set speed speed - deceleration-to-stop
      ]
      [
        set speed speed + gl_accelerator
      ]
    ]
```

```
    [
      set speed speed + gl_accelerator
    ]
    if (speed > gl_speed-maxright) [set speed gl_speed-maxright]
    if (speed < gl_speed-min) [set speed gl_speed-min]

;    plot [speed] of gl_my-car * 360000 /  5280 ; convert from feet per tick to mph
    fd speed
    set droundabout droundabout + speed
  ]
  [
;    plot [speed] of gl_my-car * 360000 /  5280; convert from feet per tick to mph
    rt 45
    fd speed
    set droundabout 0
    readjustcordinate
  ]
end

to gostraightleft
;  if [ pcolor ] of patch-ahead 5 != grey
  if droundabout = 0 ; begin roundabout left turn or go straight
  [
    set gl_cars-count gl_cars-count + 1
    set speed gl_speed-maxcircle
    set droundabout 0.00001 ; start roundabout
    rt 66.422
  ]
  ifelse ((color = sky and droundabout < gl_distance-left ) or (color = green and droundabout <
gl_distance-straight ))
  [
    let carsinroundabout other cars with [distancexy 0 0 < 15]
    fd speed
    lt speed * gl_degree-per-foot
    set droundabout droundabout + speed
    let d distancexy 0 0 / 14.9999
    setxy (xcor / d) (ycor / d)
  ]
  [
;    plot [speed] of gl_my-car * 360000 /  5280; convert from feet per tick to mph
    rt 66.422
    fd speed
    set droundabout 0
    readjustcordinate
  ]
end
```

```
to gomainroad
  let myhd heading
  let dfromcenter (distancexy 0 0)
  let speed1 speed; save the initial speed for later use
  if ( dfromcenter > 15)
  [
    let dseparation gl_minseparation + speed * gl_reactiontime
    let cars-same-direction other cars with [heading = myhd]
    ifelse any? cars-same-direction
    [
      let cars-ahead other cars-same-direction with [(distance myself) != 0 and (towards myself) != myhd]
      ifelse any? cars-ahead
      [
        let car-nearest (min-one-of cars-ahead [distance myself])
        let dtocar-nearest distance car-nearest
        let vofcar-nearest [speed] of car-nearest
        ifelse dtocar-nearest < dseparation
        [
          if speed >= vofcar-nearest
          [
            set speed speed - gl_decelerator
          ]
          if dtocar-nearest < gl_minseparation
          [
            set speed speed - gl_decelerator
            if speed >= vofcar-nearest
            [ set speed vofcar-nearest - gl_decelerator ]
          ]
        ]
        [ set speed speed + gl_accelerator ] ; end car-nearest
      ]
      [ set speed speed + gl_accelerator ] ; end cars-ahead
    ]
    [ set speed speed + gl_accelerator ] ; end cars-same-direction
  ] ; end distancexy 0 0
;  if ( (distancexy 0 0) > 15 and (distancexy 0 0) < 16.7)
 ; [
  ;  let cars-45degree cars with [(subtract-headings heading myhd = -45) ]
;  ]
  if ( dfromcenter > 25 and dfromcenter < 50 )
  [
    let d dfromcenter - 25
    let t d / speed
    let cars-inroundabout cars with [(distancexy 0 0) <= 15]
    let cars-ininterval cars-inroundabout with [(subtract-headings heading myhd > 0 )]
    if any? cars-ininterval
    [
```

```
      let deceleration-to-stop (speed ^ 2 / (2 * d))
      set speed speed - deceleration-to-stop
    ]

  ]
  let vmax gl_speed-max
  if (dfromcenter < gl_distanceto00)
  [
    let speed2 speed; save the just calculated speed
    set speed speed1; restore the initial speed
    ifelse color = red
    [
      let d sqrt ( dfromcenter ^ 2 - 36) - 31 + gl_distance-right
      let t d / speed
      let cars-perpendicular cars with [(subtract-headings heading myhd = 90) and (subtract-headings
(towards myself) myhd < 0)]
      let cars-ininterval cars-perpendicular with [(distancexy 0 0) + speed * t > 15 and (distancexy 0 0)  +
speed * t < 46.39]
      ifelse any? cars-ininterval
      [
        let deceleration-to-stop (speed ^ 2 / (2 * (d - 12)))
        set speed speed - deceleration-to-stop
      ]
      [
        set speed speed + gl_accelerator
      ]
      if (speed > speed2) [set speed speed2]
    ]
    [
      set speed speed2
    ]
    let dfromcircle sqrt(dfromcenter ^ 2 - 36) - gl_side ; gl_side = sqrt(15^2+6^2)
    set vmax sqrt(gl_speed-max ^ 2 - 2 * gl_decelerator * (gl_slowdown - dfromcircle)) ; gl_slowdown
= (v^2 - v0^2)/(2a)
  ]
  if vmax > gl_speed-max
  [
    set vmax gl_speed-max
  ]
  if speed > vmax
  [
    set speed vmax
  ]
  if speed < gl_speed-min
  [
    set speed gl_speed-min
  ]
```

```
  forward speed
end

to readjustcordinate
  if (heading > 355 or heading < 5)
  [
    set heading 0
    set xcor 6
  ]
  if (heading > 85 and heading < 95)
  [
    set heading 90
    set ycor -6
  ]
  if (heading > 175 and heading < 185)
  [
    set heading 180
    set xcor -6
  ]
  if (heading > 265 and heading < 275)
  [
    set heading 270
    set ycor 6
  ]
end
```

# Code for Roundabout without right turn lanes

```
globals [ gl_my-car gl_cars-count gl_accelerator gl_decelerator gl_speed-max gl_speed-min
  gl_speed-maxcircle gl_distance-straight gl_distance-left gl_distance-right gl_degree-per-foot
  gl_side gl_slowdown gl_straightdistance gl_distanceto00 gl_minseparation gl_reactiontime]
breed [cars car]
turtles-own [ speed droundabout ]

to setup
  clear-all
  set gl_cars-count 0
  ; 100 ticks per second
  set gl_accelerator 0.001 ; 10 ft per second per second
  set gl_decelerator 0.001 ; 10 ft per second per second
  set gl_speed-max  0.5133333333333 ; 35mph, 51.3333 ft per second
  set gl_speed-min  1.e-9 ; to prevent divided by 0
  set gl_speed-maxcircle  gl_speed-max * 0.5  ; half normal speed
  set gl_distance-straight 34.77
  set gl_distance-left 58.34
  set gl_distance-right 11.22
  set gl_degree-per-foot 3.81972
  set gl_side sqrt (15 ^ 2 - 6 ^ 2)
  set gl_slowdown (gl_speed-max ^ 2 - gl_speed-maxcircle ^ 2) / (2 * (gl_decelerator))
  set gl_straightdistance gl_slowdown + gl_side
  set gl_distanceto00 sqrt (gl_straightdistance ^ 2 + 6 ^ 2)
  set gl_minseparation 15; assume car lenght 10' & distance between cars 5'
  set gl_reactiontime 100; 100 ticks or 1 second reaction time

  create-cars NumberCars
  [
    set color (random 3 * 40 + 15) ; 15=red=go right, 55=green=go straight, 95=sky=go left
    set shape "car"
    set size 5
    set speed gl_speed-max - random-float .1
    ifelse (color = sky ) [set droundabout ( gl_distance-left)]
    [
      ifelse (color = green ) [set droundabout ( gl_distance-straight)]
      [set droundabout gl_distance-right]
    ]
    distribute-cars
  ]

  set gl_my-car one-of cars
  watch gl_my-car
  ask patches
  [
```

```
    if (pxcor ^ 2 + pycor ^ 2 < 20 ^ 2) and (pxcor ^ 2 + pycor ^ 2 > 10 ^ 2)
    [
    set pcolor grey
    ]
    if abs (pxcor) > 1 and abs (pxcor) < 11 and (pxcor ^ 2 + pycor ^ 2 > 10 ^ 2)
    [
      set pcolor grey
    ]
    if abs (pycor) > 1 and abs (pycor) < 11 and (pxcor ^ 2 + pycor ^ 2 > 10 ^ 2)
    [
      set pcolor grey
    ]
  ]
end

to distribute-cars  ;; procedure
    set heading random 4 * 90
    if (heading = 0)
    [setxy 6 (38 + random (max-pycor - 38)) * (2 * random 2 - 1)]
    if (heading = 90)
    [setxy ((38 + random (max-pxcor - 38)) * (2 * random 2 - 1)) -6]
    if (heading = 180)
    [setxy -6 ((38 + random (max-pycor - 38)) * (2 * random 2 - 1))]
    if (heading = 270)
    [setxy ((38 + random (max-pxcor - 38)) * (2 * random 2 - 1)) 6]
    if any? other turtles-here
    [distribute-cars ]
end

to go
  ask cars [goall]
  tick
  if ticks > 30000 [stop]
end

to goall
  let myhd heading
  ifelse distancexy 0 0 <= 15
  [
    goroundabout
  ]
  [
    gomainroad
  ]
  plot [speed] of gl_my-car * 360000 /  5280 ; convert from feet per tick to mph
end
```

```
to goroundabout
  let myhd heading
  if ((color = sky and droundabout = ( gl_distance-left) )
    or (color = green and droundabout = ( gl_distance-straight ))
    or (color = red and droundabout = ( gl_distance-right )))
  [ ; begin roundabout
    set gl_cars-count gl_cars-count + 1
    set droundabout droundabout - 0.00001 ; start roundabout
    rt 66.422
  ]

  ifelse droundabout > 0
  [ ; on circle
    let cars-inroundabout other cars with [distancexy 0 0 < 15]
    let cars-ahead cars-inroundabout with [(subtract-headings heading myhd > -90)
      and (subtract-headings heading myhd < 0)]
    ifelse any? cars-ahead
    [
      let car-nearest (min-one-of cars-ahead [distance myself])
      ifelse (subtract-headings [heading] of car-nearest myhd > -52.296 and
        speed > [speed] of car-nearest) ; 52.296 degrees = 15 ft
      [set speed [speed] of car-nearest - gl_decelerator] ; car closer than 15 ft
      [set speed speed - gl_decelerator] ; car farther than 15 ft
    ]
    [set speed speed + gl_accelerator] ; no cars ahead so accelerate

    if droundabout < 15 ; ready to exit roundabout and have to watch for cars on main road
    [
      let cars-onmainroad-nearby other cars with [distancexy 0 0 > 15
        and (distance myself) < 15 and subtract-headings heading myhd > 0]
      if any? cars-onmainroad-nearby
      [
        let car-onmainroad-nearest (min-one-of cars-onmainroad-nearby [distance myself])
        set speed [speed] of car-onmainroad-nearest - gl_decelerator
      ]
    ]

    if speed < gl_speed-min [set speed gl_speed-min]
    if speed > gl_speed-maxcircle [set speed gl_speed-maxcircle]
    fd speed
    lt speed * gl_degree-per-foot
    set droundabout droundabout - speed
    let d distancexy 0 0 / 14.9999
    setxy (xcor / d) (ycor / d)
  ]

  [ ; exit roundabout
```

```
    rt 66.422
    fd speed
    ifelse (color = sky )
    [
      set droundabout ( gl_distance-left)
    ]
    [
      ifelse (color = green )
      [
        set droundabout ( gl_distance-straight)
      ]
      [
        set droundabout ( gl_distance-right) ; red, turn right
      ]
    ]
    readjustcordinate
  ]
end

to gomainroad
  let myhd heading
  let dfromcenter (distancexy 0 0)
  let speed1 speed; save the initial speed for later use
  if ( dfromcenter > 15)
  [
    let dseparation gl_minseparation + speed * gl_reactiontime
    let cars-same-direction other cars with [heading = myhd]
    ifelse any? cars-same-direction
    [
      let cars-ahead other cars-same-direction with [(distance myself) != 0 and (towards myself) != myhd]
      ifelse any? cars-ahead
      [
        let car-nearest (min-one-of cars-ahead [distance myself])
        let dtocar-nearest distance car-nearest
        let vofcar-nearest [speed] of car-nearest
        ifelse dtocar-nearest < dseparation
        [
          if speed >= vofcar-nearest
          [
            set speed speed - gl_decelerator
          ]
          if dtocar-nearest < gl_minseparation
          [
            set speed speed - gl_decelerator
            if speed >= vofcar-nearest
            [ set speed vofcar-nearest - gl_decelerator ]
          ]
```

```
        ]
        [ set speed speed + gl_accelerator ] ; end car-nearest
      ]
      [ set speed speed + gl_accelerator ] ; end cars-ahead
    ]
    [ set speed speed + gl_accelerator ] ; end cars-same-direction
  ] ; end distancexy 0 0

  if ( dfromcenter > 25 and dfromcenter < 50 and subtract-headings myhd (towardsxy 0 0) < 90)
  [
    let myhdatcircle myhd + 66.422
    let dtocircle dfromcenter - 15
    let dtostop dfromcenter - 25
    let t (2 * dtocircle / (speed + gl_speed-maxcircle)) ; t is the time to circle
    let cars-inroundabout cars with [(distancexy 0 0) <= 15]
    let cars-ininterval cars-inroundabout with [(t * speed < droundabout ) or
      (t * (speed + gl_speed-maxcircle) / 2 < droundabout)]
    let cars-inrange cars-ininterval with [(
      (subtract-headings (heading + (t * speed) * 3.82) myhdatcircle) < 52.296 and
      (subtract-headings (heading + (t * speed) * 3.82) myhdatcircle) > -52.296) or
      ((subtract-headings (heading + (t * (speed + gl_speed-maxcircle) / 2) * 3.82) myhdatcircle) <
52.296 and
      (subtract-headings (heading +(t * (speed + gl_speed-maxcircle) / 2) * 3.82) myhdatcircle) > -
52.296)]
    if any? cars-inrange or count cars with [(distancexy 0 0) <= 25] > 3
    [
      let deceleration-to-stop (speed ^ 2 / (2 * dtostop))
      set speed speed - deceleration-to-stop
    ]

  ]

  if speed > gl_speed-max [ set speed gl_speed-max ]
  if speed < gl_speed-min [ set speed gl_speed-min ]
  forward speed
end

to readjustcordinate
  if (heading > 355 or heading < 5)
  [
    set heading 0
    set xcor 6
  ]
  if (heading > 85 and heading < 95)
  [
    set heading 90
    set ycor -6
```

```
  ]
  if (heading > 175 and heading < 185)
  [
    set heading 180
    set xcor -6
  ]
  if (heading > 265 and heading < 275)
  [
    set heading 270
    set ycor 6
  ]
end
```