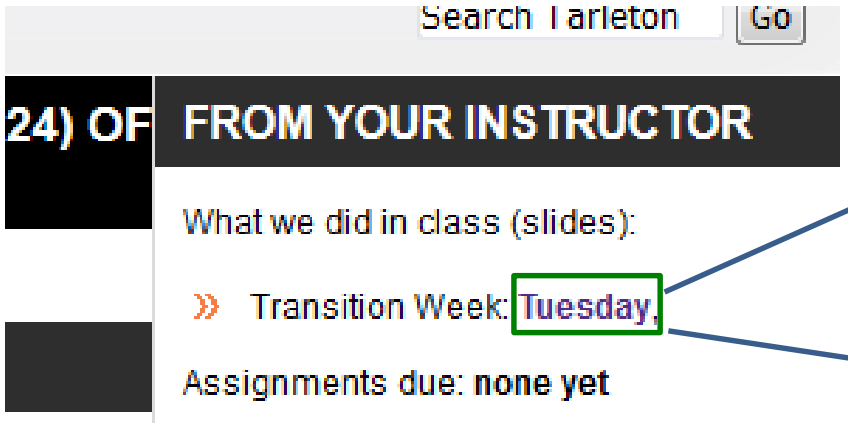
Welcome to

Computer Science!

Dr. Mircea **Agapie** Office: SCIENCE **213-C** **agapie@tarleton.edu**

Downloading documents from the course webpage



Right-click here and, from the drop-down menu, choose **Save Link As …**

Then save on your **T** drive.

QUIZ

- What is Computer Science?

QUIZ

- What is an algorithm?

QUIZ

Use numerical **algortihms** to quickly calculate in your head:

- 35 x 9 35 x 11
- 42 x 5 42 / 5
- 1.7 x 4 1.7 x 16

QUIZ

Find an **algorithm** to solve the "heavy-medium-light" problem:

- There are 3 people, whom we shall call "heavy", "medium", and "light", because they weigh 300, 200, and 100 lbs, respectively.
- They need to traverse a river with a boat that can only carry 300 lbs.
- (The boat cannot travel over the river empty.)

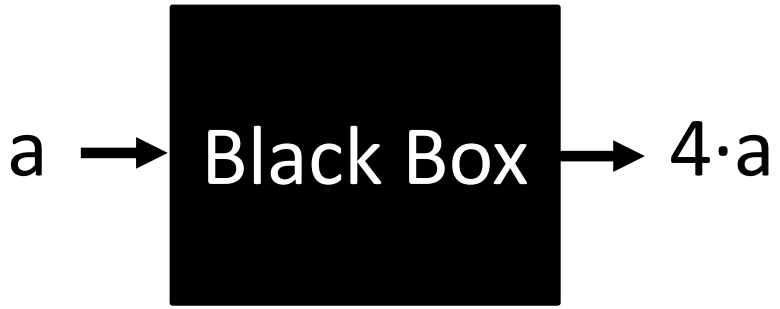
A more difficult algorithm

Find an **algorithm** to solve the "missionaries and cannibals" problem:

https://en.wikipedia.org/wiki/Missionaries_and_cannibals_problem

(Challenge: Try finding a solution yourself!)

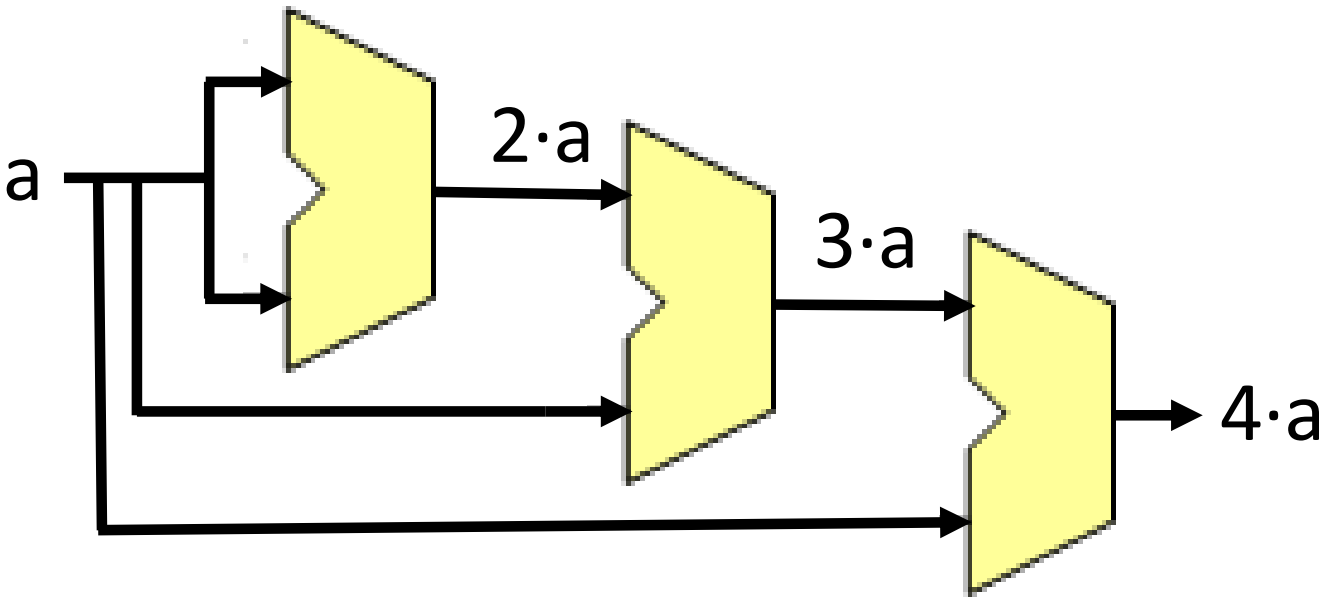
Algorithm: Multiplication by 4



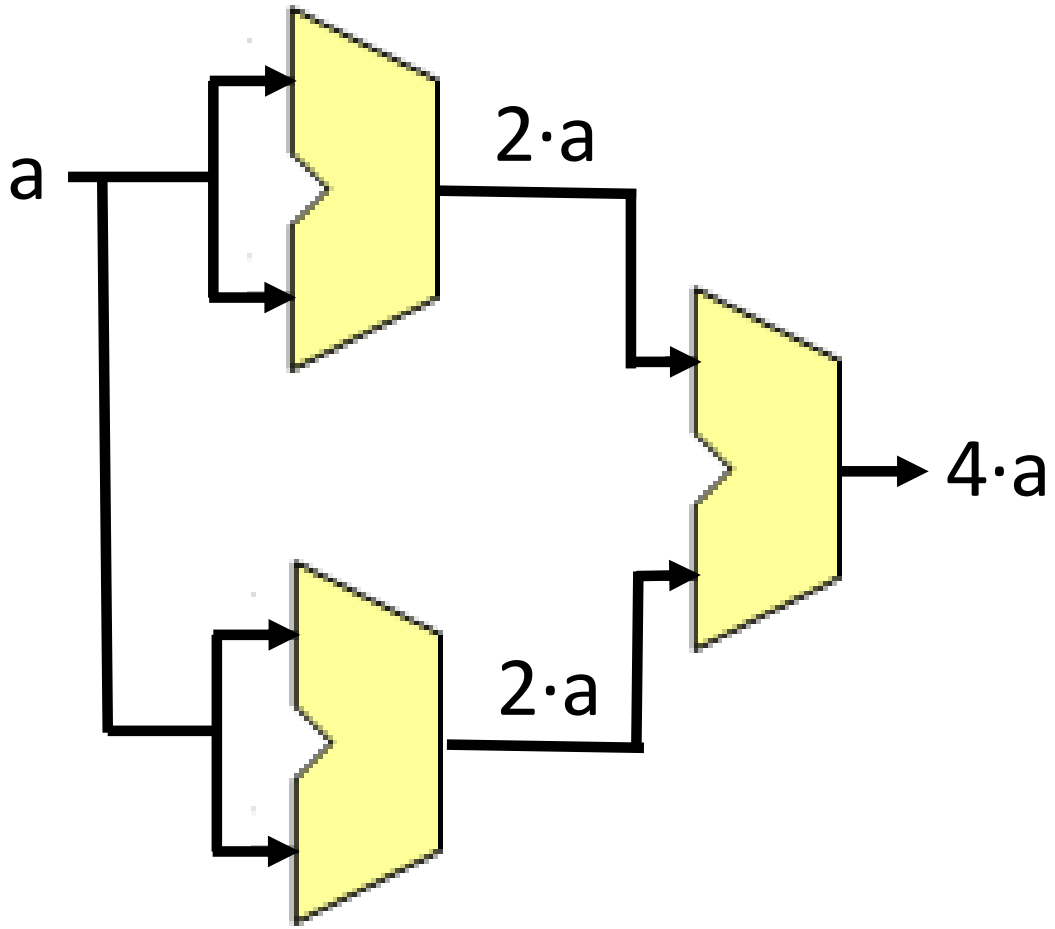
Assigned last time!

Build a circuit with <u>one</u> input a, that produces **4·a** at the output.

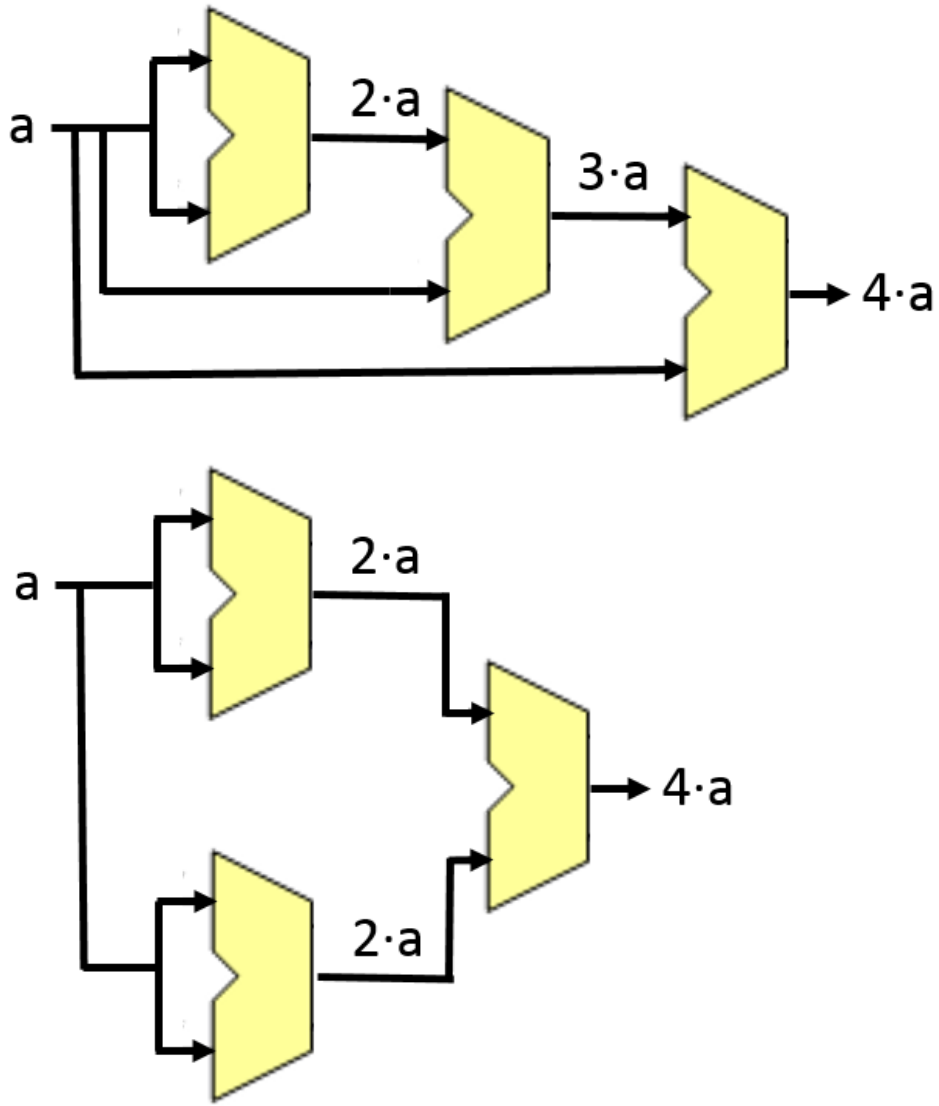
Solution 1



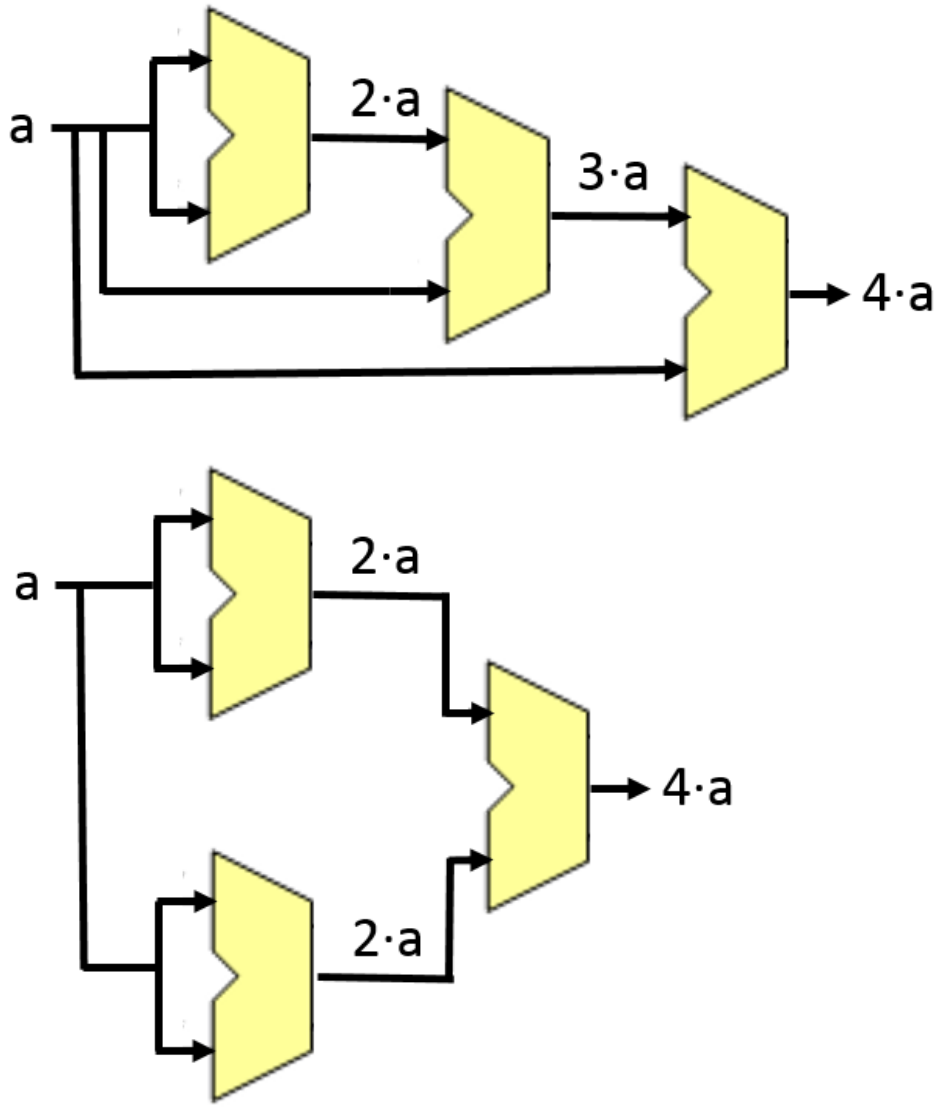
Solution 2



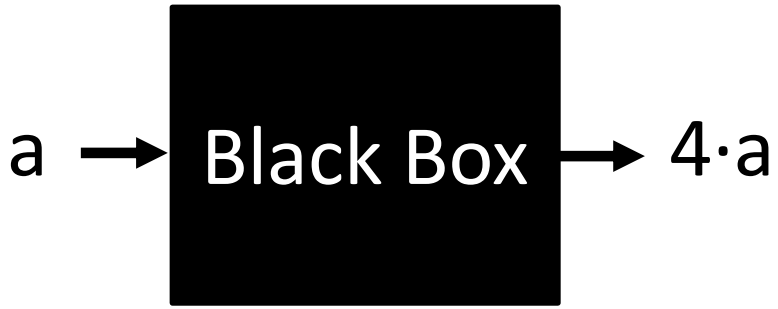
Let’s compare the two implementations …
Which one do you think is “better”?



We just **designed** and **analyzed** algorithms!!



Algorithm: Multiplication by 4



Build the circuit with only two adders!

Solve in notebook for next time!

Most algorithms are application-specific

- Playing chess
- Planning elevator motion
- Recognizing objects in digital images
- Finding information in databases
- Routing packets in the Internet
- Moving robotic arms
- Driving self-driving cars
- Matching DNA sequences
- Etc., etc., etc.

… although there are also may algorithms with wide applicability over multiple domains

- Searching
- Sorting
- Hashing
- Finding shortest paths
- Genetic Algorithms
- Neural Networks
- Etc., etc., etc.

AI vs. GAI

Robots!

ABB Fanta challenge:

- http://www.youtube.com/watch?v=PSKdHsqtok0&feature=related
- http://www.youtube.com/watch?v=SOESSCXGhFo&feature=related

BMW i3 factory:

- https://www.youtube.com/watch?v=pa5_tudyAF8

Robots!

Big Dog: http://www.youtube.com/watch?v=cNZPRsrwumQ

Ping Pong: http://www.youtube.com/watch?v=t_qN3dgYGqE (2011)

http://on.aol.com/video/ping-pong-robot-518456199 (2014)

Surprise: https://www.youtube.com/watch?v=ub4s984sL0A ☺

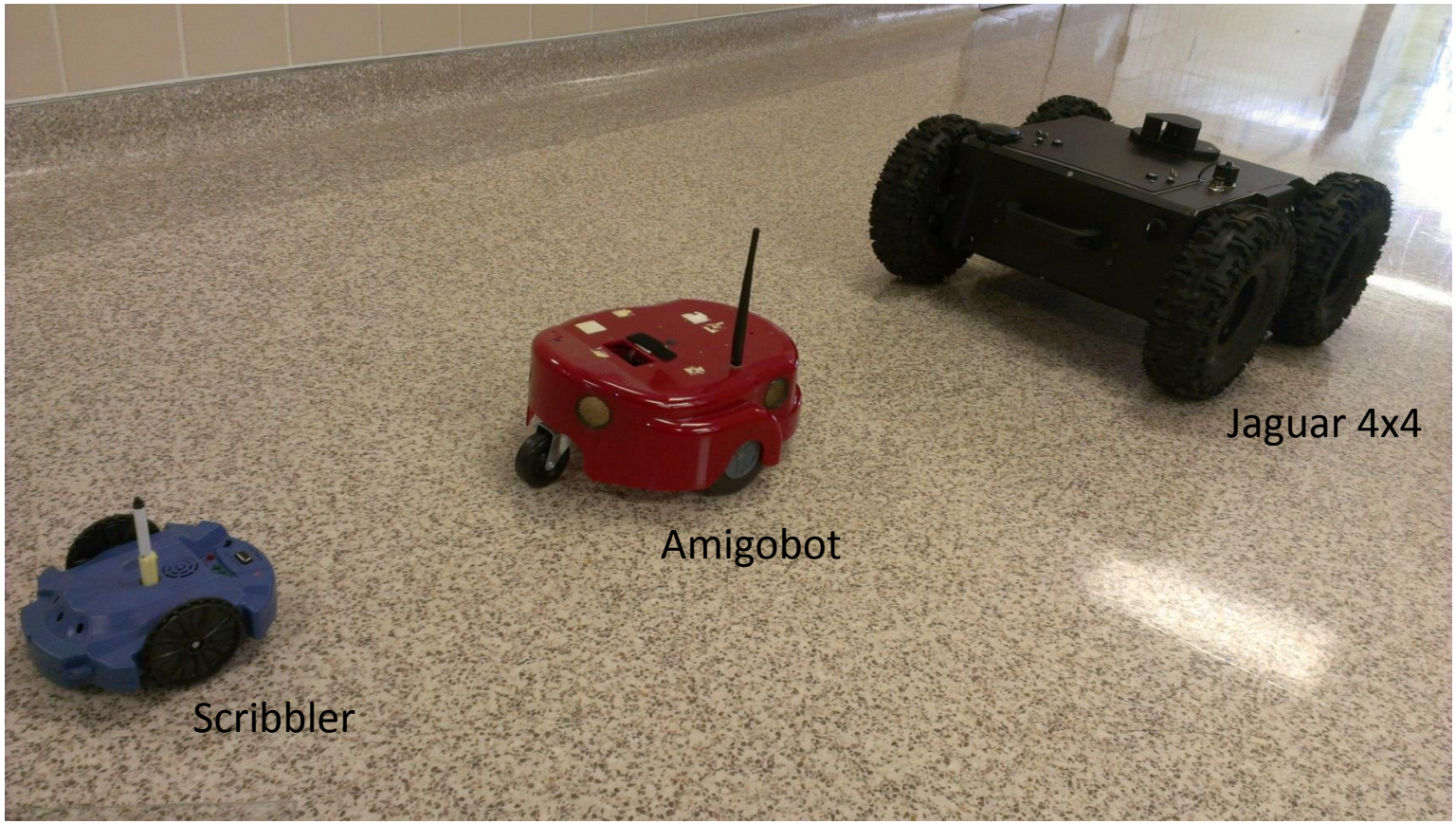
Robots at Tarleton: Autonomous and mobile

Jerry Barnett (graduated in 2014)

Robots at Tarleton

Robotics Summer Camp for high-school students (2014)

Robots at Tarleton



Robots at Tarleton

Colby Larue worked on the Jaguar in the Summer of 2015 and presented this poster at the TAMUS student research symposium:

Behavior-Based Navigation for Autonomous Mobile Robot

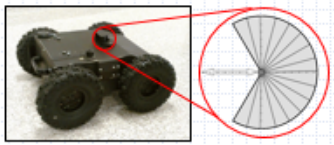
Colby LaRue Mentor: Dr. **Mircea Agapie**

Dept. of Engineering & Computer Science, Tarleton State University

Abstract

The goal of this project was to design and implement a behavior-based algorithm for robot navigation in a simple, static environment, such as an uncluttered corridor or hallway, without moving objects. For actuation, the robot has a 4-wheel differential drive, and for sensing it has a top-mounted Hokuyo laser range sensor that is used to scan the environment. The main control program runs on a stationary PC, which communicates with the robot via wireless Ethernet (Wi-Fi); the communication is two-way, with motion commands going from PC to robot, and sensor readings going from robot to PC. The final algorithm consists of five "behaviors" and a decision mechanism that switches among them according to what the robot is "seeing" with the laser sensor. The robot is able to veer away from lateral obstacles; when encountering a frontal obstacle, it stops and spins, searching for a new direction, and then resumes navigation. The manufacturer's API (Application Program Interface) was used to implement the navigation algorithm in C++, under the .NET framework.

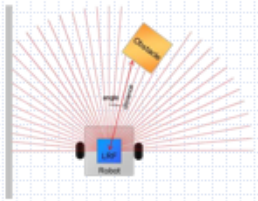
Jaguar 4x4 Robot



Manufactured by Dr. Robot, Inc.

Jaguar-4x4-wheel Mobile Robotic Platform is designed for indoor and outdoor operation requiring higher ground clearance and faster maneuverability. Jaguar-4x4-wheel platform is a wheeled version of the Jaguar-Lite platform. Jaguar-4x4-wheel is driven by four powerful (80W) motors, one for each wheel. Jaguar-4x4-wheel platform is rugged, light weight (< 20Kg), fast (max 11 km/hr.), with high ground clearance (88mm), compact, weather and water resistant. It is designed for tough terrains and capable of running over vertical step up to 155mm and climbing up low rise stairs (up to 110mm step). Jaguar-4x4-wheel is fully wirelessly 802.11N connected. It can integrate an outdoor GPS and 9 DOF IMU (Gyro/Accelerometer/Compass) for completely autonomous navigation. The integrated high resolution video/audio and laser scanner provide remote operator detail information of the surrounding.

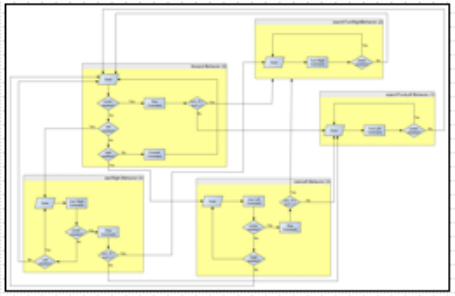
Laser Range Finder



URG-04LX-UG01 is a laser sensor for area scanning. The light source of the sensor is infrared laser of wavelength 785nm with laser class 1 safety. Scan area is 240¼ semicircle with maximum radius 4000mm. Pitch angle is 0.36¼ and sensor outputs the distance measured at every point (683 steps). Laser beam diameter is less than 20mm at 2000mm with maximum divergence 40mm at 4000mm.

Principle of distance measurement is based on calculation of the phase difference, due to which it is possible to obtain stable measurement with minimum influence from object's color and reflectance.

Navigation Algorithm



The navigation algorithm contains the five behaviors: forward, searchTurnRight, searchTurnLeft, veerRight, veerLeft. They are implemented in C++ with a switch statement that decides which behavior to activate next, as seen below:

Sample C++ code

```
while(timerCtrValue1 < 3000){ //**********3000 = 1 min; 1200 = 2 min; 3000 = 5 min**********
    switch (behavior){
        case 0: // 0 is no violation
            behavior = ClaguarMotionControlDll::forward(); //robot moves forward until an obstacle is reached
            break;
        case 1: // 1 is center violation and will turn left
            behavior = ClaguarMotionControlDll::searchTurnLeft();
            break;
        case 2: // 2 is center violation and will turn right
            behavior = ClaguarMotionControlDll::searchTurnRight();
            break;
        case 3: // 3 is left violation
            behavior = ClaguarMotionControlDll::veerRight();
            break;
        case 4: // 4 is right violation
            behavior = ClaguarMotionControlDll::veerLeft();
            break;
        default:
            behavior = ClaguarMotionControlDll::stop();
    }
}
```

The **while** loop shown above constitutes the main loop of our AI program. Each behavior function returns a value between 0 and 4 that is used in the next iteration of the loop to control the switch statement to select the next behavior.

```
if(min_C < CenterStopDistance){
        m_Jaguar.DcMotorPwmNowTimeCtr(3, (short)16383);
        m_Jaguar.DcMotorPwmNowTimeCtr(4, (short)16383);
        if(sum_L > sum_R)
            return 1;
        else
            return 2;
}
else if(min_L < SideStopDistance)
        return 3; //3 is left violation: veerRight()
else if(min_R < SideStopDistance)
        return 4; //4 is right violation: veerLeft()
else{
    m_Jaguar.DcMotorPwmNowCtr(3, (short)forwardPWM, 500);
    return 0;
}
```

This code is part of the forward behavior function. It checks the minimum value in the center section of the laser readings. If less than a certain threshold, the robot is given stop commands on both "channels": channel 3 is movement in a straight line, and channel 4 is spinning.

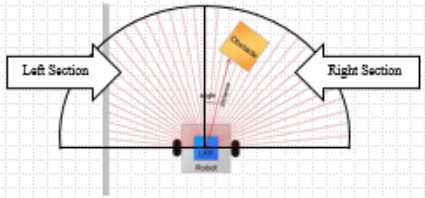
If the minimum value in the left or right sections is less than the side threshold, the veering behaviors are activated. Otherwise the robot keeps moving forward. Note in each case the **return** statement with an appropriate integer value (1, 2, 3, 4, 0), which will be used in the next loop iteration.

The program makes use of several numeric constants that express distances in the environment or PWM commands for the motors, e.g. 16383 instructs a motor to not move:

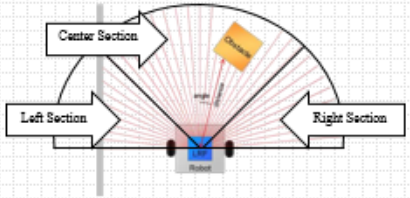
```
int forwardPWM = 16383 - 1800;
long CenterStopDistance = 900;
long SideStopDistance = 600;
bool stop = false;
long min_R = 10000, min_C = 10000, min_L = 10000;
long sum_R = 0, sum_L = 0;
```

Constants defined in behavior functions

Partitioning the Sensory Space



The laser readings are divided into a left and a right section in order to make a turn decision.



The laser readings are partitioned into a left, center, and right section for detection of side vs. frontal obstacles.

The three and two partitions shown above are implemented differently in the code: the former are based on finding the **minima** of the left, right, and center sections, whereas the latter are based on the sum (numerical integral) of the left and right sections, as seen in the code below:

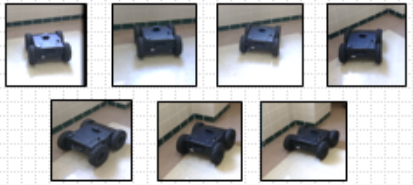
```
int i = 0;
for(it = filtered.begin(); it != filtered.end(); ++it, i++)
{
    if(i < 275)
        if(*it < min_R)
            min_R = *it; //calculates the minimum value of the right section
    if(i > 274 && i < 400)
        if( *it < min_C)
            min_C = *it;//calculates the minimum value of the center section
    if(i > 399)
        if (*it < min_L)
            min_L = *it;//calculates the minimum value of the left section

    if(i<338)
        sum_R += *it;//calculates the sum of the right section
    else
        sum_L += *it;//calculates the sum of the left section
}
```

Experiments

All experiments have been conducted in an uncluttered environment, containing only static obstacles, as shown below:

Shown here is the Jaguar 4x4 Robot performing a veerRight behavior to avoid collision with the wall.



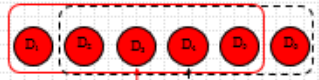
Shown here is the robot moving toward the wall, stopping, turning until no frontal object is detected, then moving forward again until a new obstacle is detected.

Data Cleaning (Denoising)

First, we iterate through the array of laser readings to check the values. If a number less than the threshold is found it will be replaced with the data points on each side of it averaged. If a number less than the threshold is found on either end of the array the next good value replaces it.

```
replaceValue = (data[i-1]+data[k])/2;
for(unsigned j = i; j < k;  j++)
    data[j] = replaceValue;
i = k + 1;
```

A windowing technique (shown below) is then used to average every set of 5 data points or array readings. This is done to correct errors in the laser readings and to ensure accuracy. A section of 5 data points is averaged; the center data point is then replaced with that averaged value. After replacing the value the 5 data point window moves over one data point.



$$D_3 = \frac{D_1+D_2+D_3+D_4+D_5}{5} \qquad D_4 = \frac{D_2+D_3+D_4+D_5+D_6}{5}$$

CONCLUSIONS

- Our work demonstrates successful integration of Artificial Intelligence and Behavior-Based navigational algorithms.
- Movement of robot is still too jerky, as robot sometimes makes unwanted turns.

FUTURE WORK

- Following a wall by keeping the distance to the wall constant
- Making a map of the environment
- Finding a target with a certain signature based on sonar or camera information – the latter involves Machine Vision.

REFERENCES

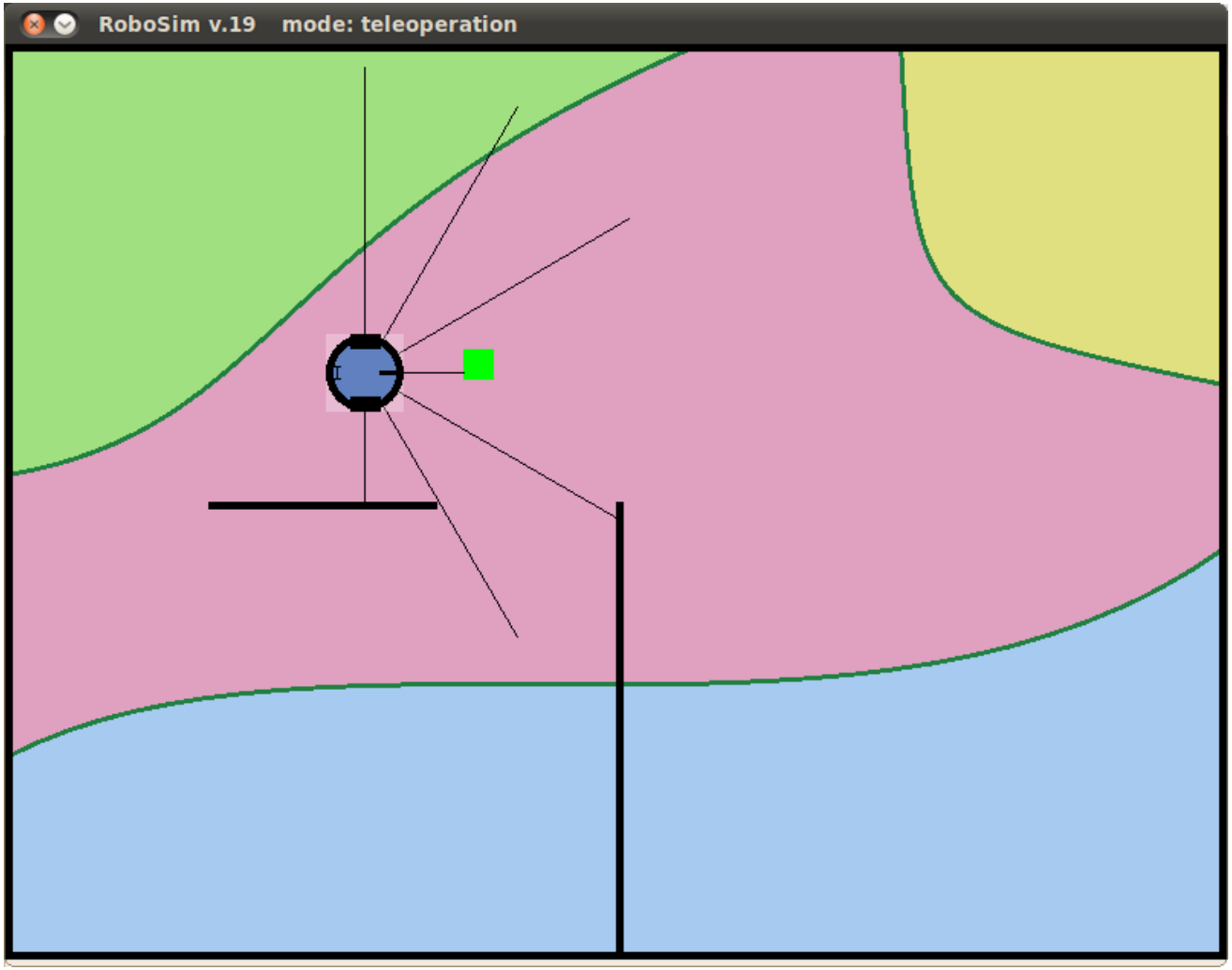
- http://jaguar.drrobot.com/specification_4x4w.asp
- https://msdn.microsoft.com/en-us/library/bb483042.aspx

For additional information please contact:
Colby LaRue colby.larue@go.tarleton.edu
Mircea Agapie agapie@tarleton.edu



*Shown here is the Jaguar 4x4 Robot performing a **veerRight** behavior to avoid collision with the wall.*

Shown here is the robot moving toward the wall, stopping, turning until no frontal object is detected, then moving forward again until a new obstacle is detected.


RoboSim v.19 mode: teleoperation

What Michael Osei did this summer

Michael is a CS sophomore who joined Tarleton in the Spring 2015 semester.

http://healthcare.tarleton.edu/public/

He built this website for the College of Business and Administration (COBA) as a summer project.

Top 10 Reasons to Major in CS

1. Computing is part of everything we do!

2. Expertise in CS enables us to solve complex, challenging problems.

3. Computing enables us to make a positive difference in the world.

4. CS graduates can pursue many types of lucrative careers.

5. Computing jobs are here to stay, regardless of where you are located.

Source: http://computingcareers.acm.org/

Top 10 Reasons to Major in CS

6. Expertise in CS helps even if our primary career choice is something else.

7. Computing offers great opportunities for creativity and innovation.

8. Computing has space for both collaborative and individual work.

9. CS is an essential part of well-rounded academic preparation.

10. Future opportunities in computing are without boundaries.

Source: http://computingcareers.acm.org/

Mythbusting:

Two popular misconceptions about Computer Science …

~~Salaries for CS professionals fall as companies turn to cheaper labor overseas~~

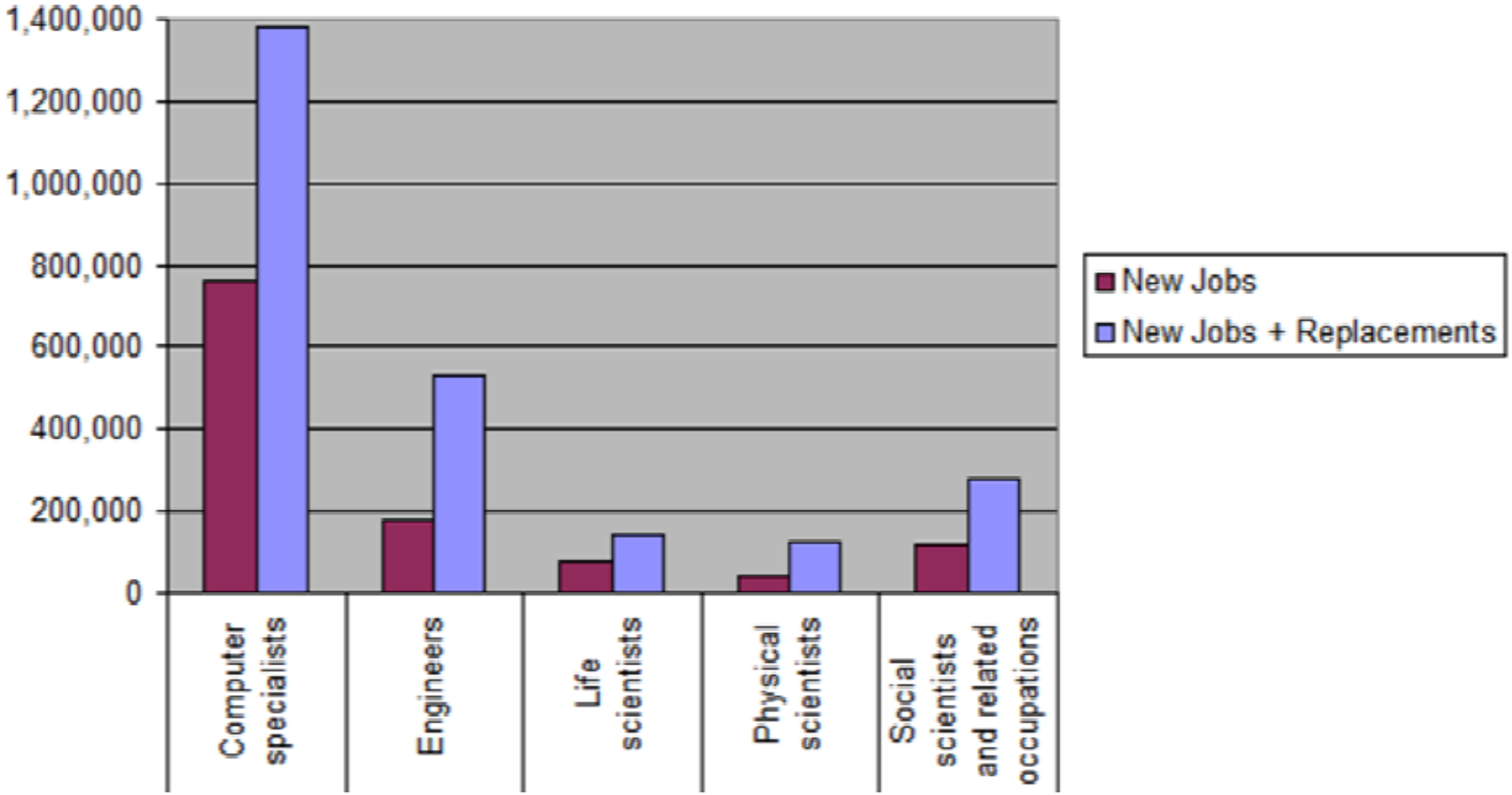
The truth:

- Offshoring does not halt the growth of CS jobs in the US because companies seek to maximize return rather than to minimize cost.
- CS developers generate far more value for their companies than they cost, even at the high salaries that such positions command in the US.

Check out the Bureau of Labor Statistics:

http://www.bls.gov/ooh/computer-and-information-technology/home.htm

Science and Technology Job Growth, 2008-2018
(Bureau of Labor Statistics)

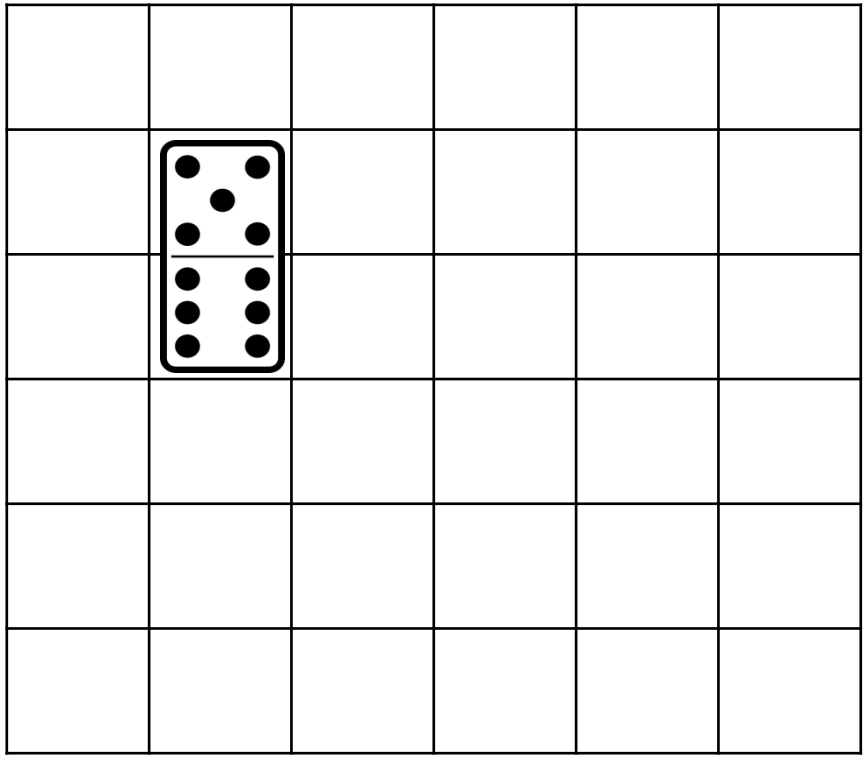


~~CS jobs are solitary and boring~~

The truth:

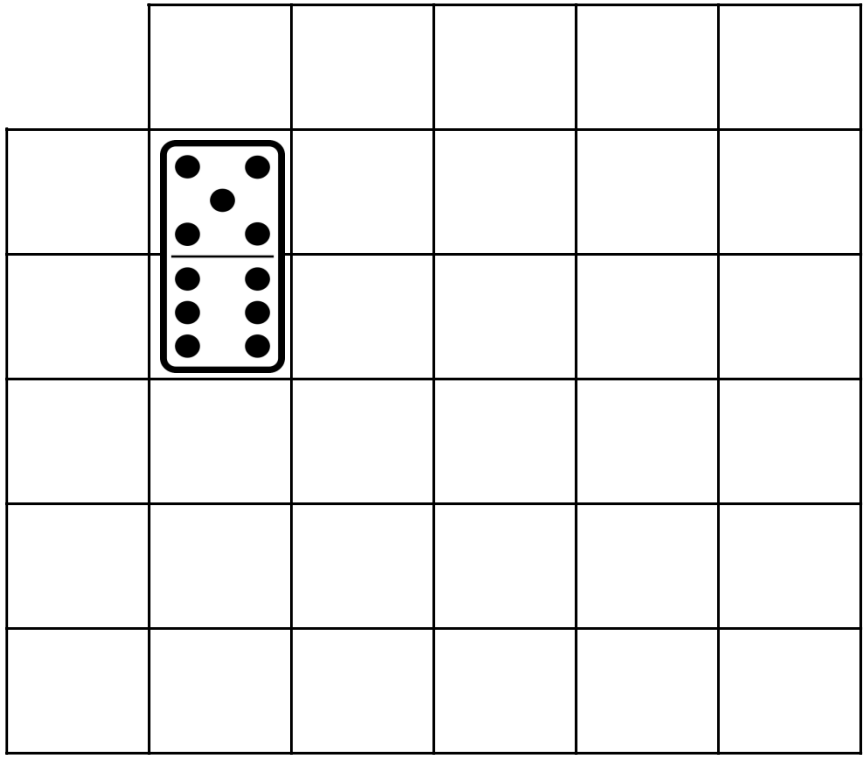
- Computing professionals hardly ever work alone. Today, building any computer system (robots, databases, networks, etc.) requires the coordinated efforts of **many people** with a wide variety of skills.
- Designing a successful product requires effective **communication** not only among the members of the development team but also with the users. Employers routinely cite good communication skills as an essential requirement for success in the field.
- CS is also a highly **creative** activity. There is very little that is mechanical about software development—if there were, those aspects would have been automated years ago.

Creative Algorithms: Tiling with Dominoes



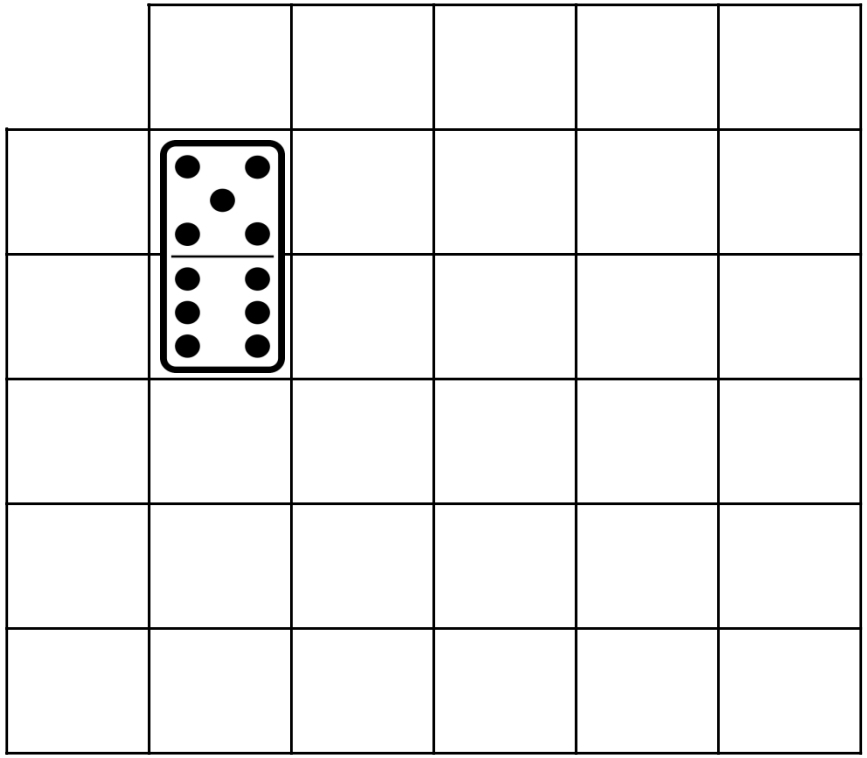
Can you cover the entire board with (non-overlapping) dominoes?

Algorithm: Tiling with Dominoes



Can you cover the entire board with (non-overlapping) dominoes?

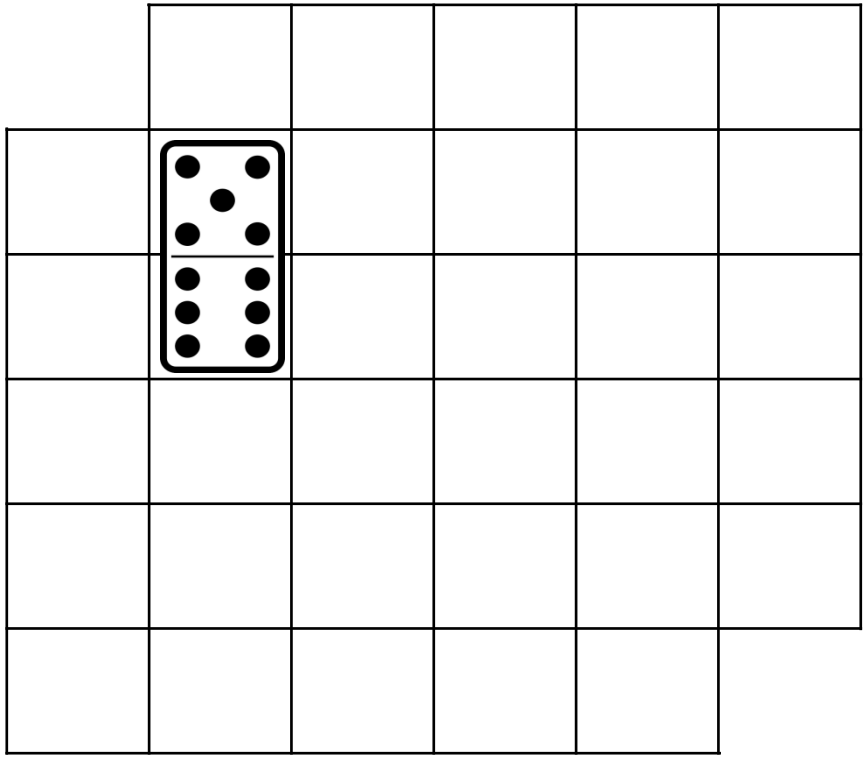
Algorithm: Tiling with Dominoes



Why not?

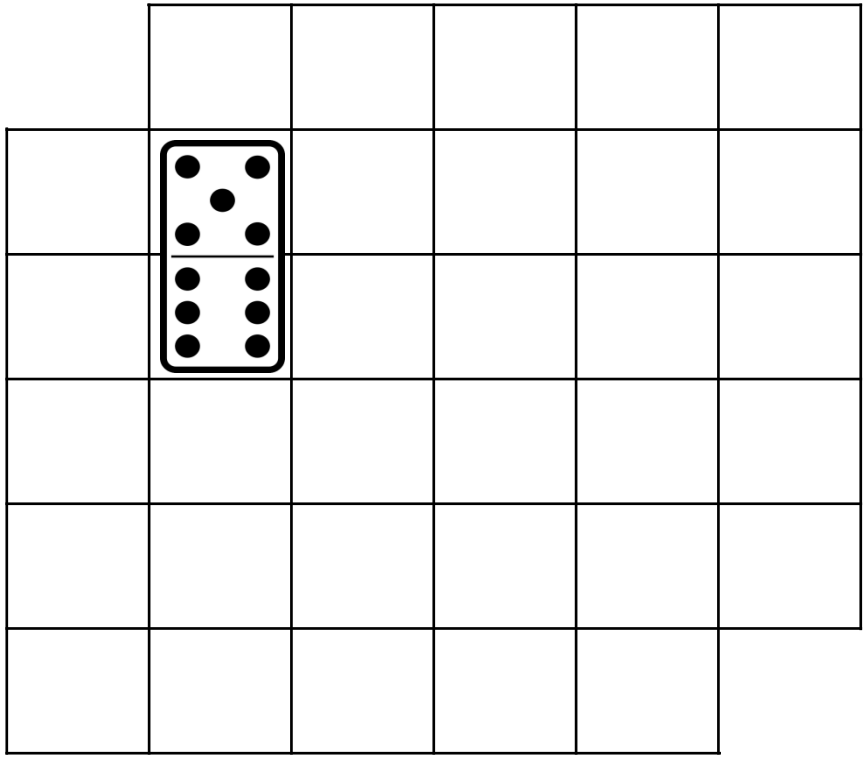
Hint: Notice an important fact about dominoes!

Algorithm: Tiling with Dominoes



Can you cover the entire board with (non-overlapping) dominoes?

Algorithm: Tiling with Dominoes



Why not?

Hint: Notice another important fact about dominoes!

Algorithm: Tiling with Dominoes