Duplo Mindstorm

Prepared by:

Cody Radabaugh

Sam Stage

Dana Clark

Tommy Trinh

Advisor:

Dr. Jamieson

Spring 2025

College of Engineering and Computing

Department of Electrical and Computer Engineering

ECE 449: Senior Design Project

Table of Contents

Introduction	**3**
Project Goal	3
Background	3
Project Research	**4**
Arduino vs Other Microcontrollers	4
Components	5
3D Modeling & Printing	6
Solution Process: Car	**7**
3D Modeling & Printing	7
Hardware/Wiring	12
Component Integration	14
Code	15
Solution Process: Crane	**17**
3D Modeling & Printing	17
Hardware/Wiring	21
Component Integration	21
Code	22
Final Results & Future Work	**24**
Proof of Concept	**26**
Conclusion	**27**
Reflection	**28**
Works Cited	**30**

Introduction

Project Goal

The goal of this project is to design a Duplo-based robot, with components specifically scaled and prototyped to integrate seamlessly with Duplo blocks. This project requires a combination of 3D printing, modeling, and hardware integration to ensure compatibility and functionality at the Duplo scale. Our focus is on integrating various sensors such as stepper motors, H-Bridge motor drivers, Arduino, IR remote, IR sensor, and power supply. These components will be housed in custom 3D-printed Duplo blocks, allowing for a variety of robot designs, similar to Mindstorms from Lego. We will document the entire design process, including the evaluation of different design choices, and provide demonstrations of various robot configurations.

Background

Lego Mindstorms is a robotics kit that combines hands-on building with block programming. Lego created this line in 1998 in collaboration with the MIT Media Lab to develop a robotics kit that would allow users, particularly children, to program their Lego robots. These kits are popular in educational settings as well, forming the basis of school programs for introduction to robotics and global competitions such as the first Lego League. The most recent iteration, the Mindstorms Robot Inventor 51515, supports programming languages like Scratch and Python through a companion app, making it accessible for users at various programming skill levels. Although Lego officially discontinued the Mindstorms line in recent years, its

influence remains, as it helped popularize STEM-focused educational products and inspired countless enthusiasts to continue innovating within the Lego community.

Duplo blocks were developed by Lego for young children and are designed to be double the size of standard Lego bricks. This makes them safer and easier for small children to handle. Duplo blocks are particularly well-suited for 3D printing and modeling robot components given the readily available parts. The larger size of Duplo blocks make ideal housing structures for the various electronic components such as sensors, motors, and microcontrollers available in the embedded system market.

Project Research

Arduino vs Other MicroControllers

We chose the Arduino Uno since it is a more suitable choice compared to other microcontroller platforms, such as those based on more powerful chips like the Raspberry Pi. The Arduino Uno, with its ATmega328P microcontroller, is optimized for simple to moderate applications, featuring an 8-bit architecture and a 16 MHz clock speed. Its 14 digital I/O pins, including six PWM outputs, provide the necessary functionality for precise motor control and analog signal simulation. Unlike more complex chips that often run full operating systems, the Arduino Uno excels in real-time control applications where low latency and real-time hardware interaction are critical. This makes Arduino ideal for controlling hardware components such as motors and sensors in our project. Additionally, its straightforward power requirements operate within a 7-12V input range and feature a built-in voltage regulator. This aligns well with our battery-powered setup, ensuring stable and efficient performance. The simplicity of the Arduino

IDE, with its support for C and C++, allows for rapid development and debugging, while its compact design and compatibility with sensor shields make hardware integration seamless. More powerful platforms would have introduced unnecessary complexity, increased power consumption, and higher resource requirements for tasks that the Arduino Uno can handle efficiently.

Components

IR Sensor

We chose the KY-022 IR receiver sensor to read signals from the IR remote. We got this piece off of Amazon for $6.99 for a 3-pack. The KY-022 is an IR receiver module used to detect signals from IR remote controls. It converts incoming IR light pulses into a digital signal that microcontrollers can read. This makes it useful for projects involving remote control input or wireless communication.

Stepper Motor/H-Bridge Motor Driver

We chose to use the stepper motor which we got on Amazon in a 6-pack for $7.99. A stepper motor is a type of motor that works in steps which allows for fine control of speed, acceleration, and position. Typical stepper motors work with a driver that controls the voltage and direction of the current that dictates the speed and direction the motor is spinning. This allows for integration in small circuits like the one we are attempting to create. It is fairly straightforward to integrate Arduino. We used the 28byj-48 stepper motor which can be purchased on Amazon for $2.96.

Servo Motor/Motor Driver

As you will see later in this report we built a crane-like robot. We used a 28byj-48 motor to drive the base of the crane and the spool. This component was inexpensive and good for our purposes. The motor driver was a ULN2003A. We used two of these to drive each motor through the Arduino. You can purchase the ULN2003A motor driver on Amazon for $2.16.

Power Supply

The battery packs supply a 12 V DC value which is needed for our motor drivers. We used the HW-688 power regulator to allow for easy turn on and off by unplugging the barrel jack of the battery pack. The battery packs are rechargeable which was good for our project because of the heavy testing we did. You can purchase the battery pack on Amazon for $16.99. We did not have to buy the power supply because we borrowed it from another group.

3D Modeling & Printing

3D Modeling

We used Tinkercad for our modeling program. Tinkercad is a web-based software by Autodesk that enables users to design 3D models. It offers a simple drag-and-drop method for creating designs without requiring prior technical skills. You can find downloadable files completed by some of their users. We were able to download an STL

duplo block file directly from Tinkercad. Next, we edited the downloaded block by inserting new shapes, to either delete or add new sections onto the block, creating our final models.

3D Printing

Bambu P1S printers work by creating objects layer by layer using a heated nozzle that melts filament which is fed through an extruder. The printer's software slices a 3D model into thin layers, which the machine then follows to build the object from the base up. The nozzle moves along the X and Y axes to shape each layer, while the bed lowers along the Z-axis as the print progresses. Features like automatic bed leveling and cooling fans help ensure consistent quality. The user uploads designs through a hard drive.

Solution Process: *Car*

3D Modeling & Printing

Given the straightforward nature of our 3D modeling requirements, we determined that Tinkercad was the most suitable software for our project. Its intuitive interface and ease of use allowed us to efficiently design and modify our Duplo-compatible components. Once the designs were finalized, we exported them as STL files. All of the design files are on our Github page [1].

Arduino Enclosure

Figure 1: Arduino Enclosure

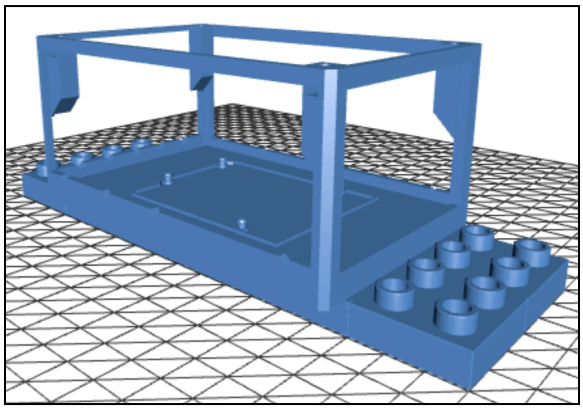


Figure 2: Arduino Enclosure Lid

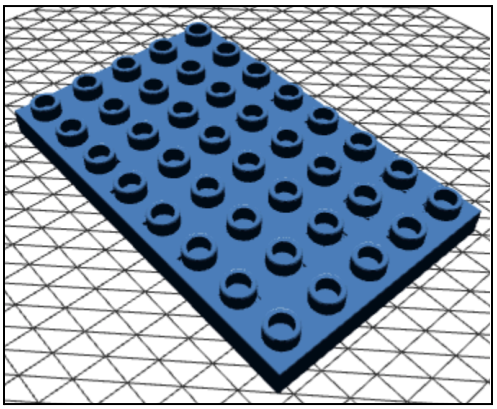


Figure 1 features an enclosure for an Arduino Uno. The design features 2 2x4 duplo pieces on each end of the enclosure. It has pegs in the middle for the arduino to mount on. Figure 2 displays the arduino enclosure lid. The lid is 5x8 and rests inside the outer framing of the top. The open walls create a lightweight enclosure. The enclosure's base could be expanded to allow for more duplo blocks on the sides.

H Bridge Enclosure

Figure 3: H Bridge Enclosure

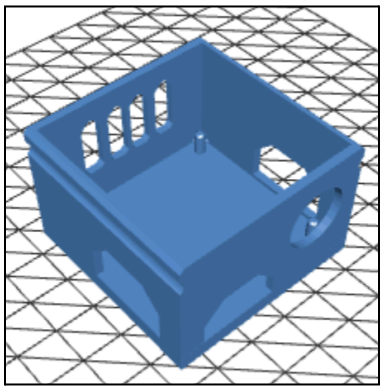


Figure 4: H Bridge Enclosure with Lid

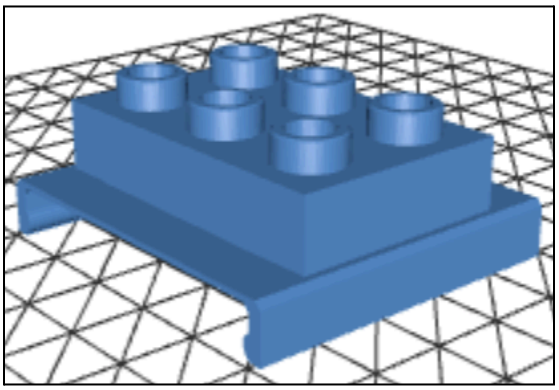


Figure 3 displays the H bridge enclosure (right). The design of the duplo block features holes for the wires to pass through and allow for the cooling of the H Bridge. Figure 4 displays the lid for the H bridge (left). The lid slides onto the H Bridge enclosure with a 2x4 duplo block on the top. This design supports efficient wiring and airflow. However, the holes restrict the wires to only be able to come out of the enclosure from one side and limit freedom.

Battery Enclosure

Figure 5: Battery Enclosure

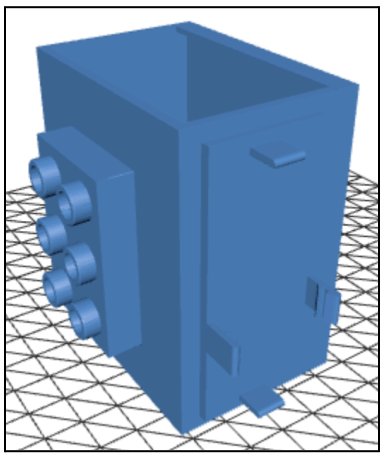


Figure 5 features an enclosure for a 12 V battery pack with a mount for a buck module. The design has a 2x4 duplo block attached to the enclosure with an open end for the battery wires to exit. The side mount holds the buck module in place, allowing the buck module to connect with the battery. The battery pack weight causes the enclosure to struggle staying attached to other duplo blocks. Designing the enclosure to have more duplo pegs can help fix this.

Motor Duplo Mount

Figure 6: Motor Mount

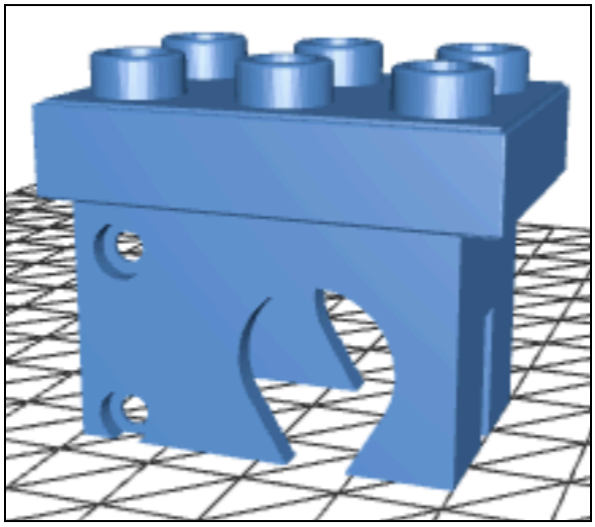


Figure 6 features an enclosure for a Motor. The block has holes for both screws and nuts to secure the motor to the mount. The 2x3 duplo on top makes it easy to connect to the bottom of the Arduino enclosure (Figure 1). This model works well with the motor and other Duplo blocks.

IR Receiver Duplo Mount

Figure 7: IR Receiver Enclosure

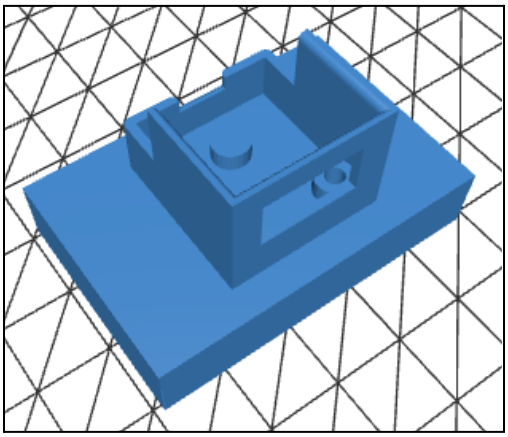


Figure 7 features an enclosure for holding the IR receiver. It is designed with a 2x3 duplo on the bottom to attach to the top of the lid easily. The open top allows for easy connection to the IR remote. This design allows a little movement for the IR receiver which can be improved in the future.

Caster Mount

Figure 8: Caster Mount

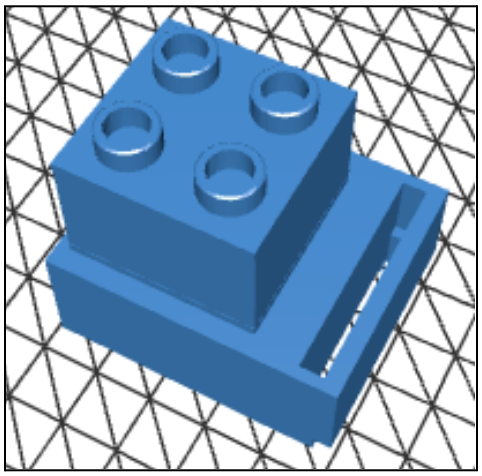


Figure 8 is a mount for the caster wheel. There is a 2x2 duplo block on top of the model. The caster wheel provides for more stability for the robot.

For the printing process, we utilized Bambu P1S printers. Using Creality's software, we converted the STL files we got from Tinkercad into the file needed to be able to print it out which is called GCODE The GCODE files were then uploaded to a drive to transfer to the 3D printers. We opted for standard black PLA filament, which was readily available to us.

Hardware/Wiring

Table 1: Car Pin Plan

Pin	**Component**
Digital 7 S , V , G	*IR Receiver S, + , -*
Digital 5 S	*H Bridge Motor 2 Enable*
Digital 6 S	*H Bridge Motor 1 Enable*
Digital 7 S	*H Bridge Motor 1 IN1*
Digital 11 S	*H Bridge Motor 1 IN2*
Digital 10 S	*H Bridge Motor 2 IN1*
Digital 9 S	*H Bridge Motor 2 IN2*

Figure 9: Arduino Uno Wiring

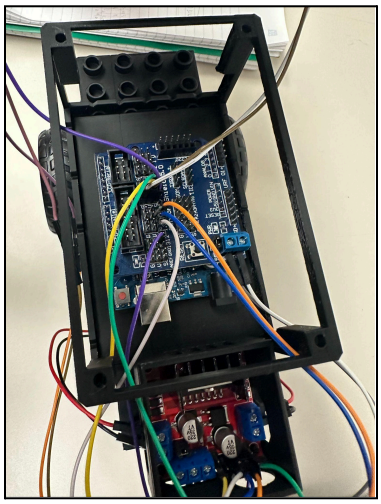


Figure 9 displays the Arduino Uno wiring in the enclosure. The wiring setup consists of multiple wires connected to the input and output pins of the Arduino board, for interfacing with the external components. The pin map can be found in Table 1.

Figure 10: H Bridge Wiring

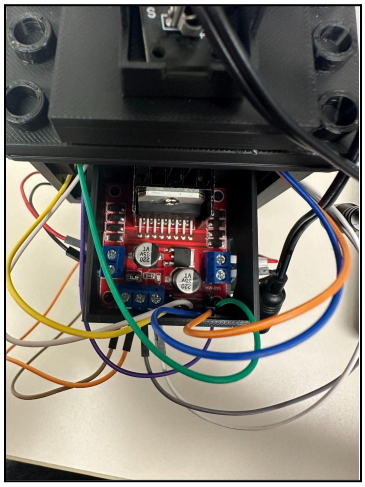


Figure 10 features the wiring for the H Bridge. which is used for controlling the direction and speed of the motors. Various wires are linked to the control signals from the Arduino, allowing it to manage the power flow to a motor in both directions. The H-Bridge allows for bidirectional motor control, enabling the connected motor to rotate forwards or backwards depending on the inputs it receives.

Figure 11: IR Receiver Wiring

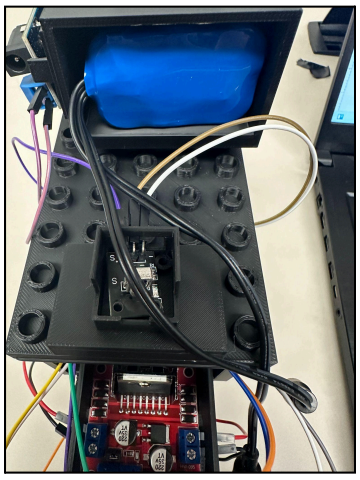


Figure 11 contains an image of the IR receiver wiring. The IR receiver is only connected to the Arduino Uno. This configuration enables the Arduino to receive and process infrared signals.

Figure 12: Buck Module Wiring

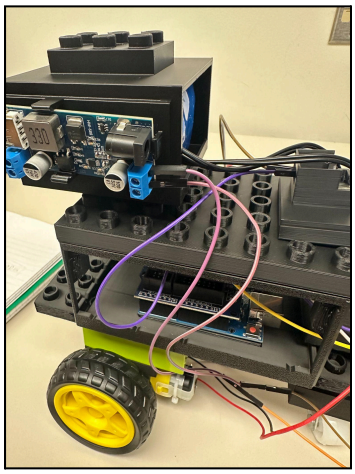


Figure 12 shows the buck module wiring. The buck converter manages the regulated power distribution to ensure the appropriate voltage levels for the connected electronics. The module acts as a switch allowing us to turn the robot on and off by unplugging the barrel jack from the battery pack.

Component Integration

Leveraging our experience from ECE 314, the Elements of Robotics course at Miami University, we integrated familiar components efficiently. The Arduino Uno was selected as the main board for its reliable voltage regulation and compatibility with the Arduino IDE. A sensor shield was added to facilitate connections with motor drivers, which control the stepper motors. These motors receive directional signals from the drivers, enabling precise forward and reverse movement. For power, we utilized a 12V battery pack and a power regulator from a previous

project. Each component was housed in a custom 3D-printed Duplo block, allowing for modularity and user creativity in assembling different robot configurations.

Code

Figure 13: Demo Code

```
// Cody Radabaugh, Sam Stage, Dana Clark, Tommy Trinh

#include <IRremote.h>

// Defining IR pin and values received from 1-Forward, 2-Backwards, and 3-Stop
// from the IR remote
#define IR_Pin 7
#define FORWARD_COMMAND 24
#define BACKWARD_COMMAND 12
#define STOP_COMMAND 94

// Initializing pins for motor driver
const int RSPD = 120;
const int LSPD = 120;
const int RWhPWMPin = 6;
const int RWhFwdPin = 11;
const int RWhBwdPin = 12;
const int LWhFwdPin = 9;
const int LWhBwdPin = 10;
const int LWhPWMPin = 5;
bool stop = true;

void setup() {
  // Configure motor driver pins as outputs
  pinMode(LWhFwdPin, OUTPUT);
  pinMode(LWhBwdPin, OUTPUT);
  pinMode(LWhPWMPin, OUTPUT);
  pinMode(RWhFwdPin, OUTPUT);
  pinMode(RWhBwdPin, OUTPUT);
  pinMode(RWhPWMPin, OUTPUT);

  // Initialize IR receiver
  IrReceiver.begin(IR_Pin, ENABLE_LED_FEEDBACK);

  // Initialize motor driver pins to low/off state
  digitalWrite(LWhFwdPin, LOW);
  digitalWrite(LWhBwdPin, LOW);
  digitalWrite(LWhPWMPin, 0);
  digitalWrite(RWhFwdPin, LOW);
  digitalWrite(RWhBwdPin, LOW);
  digitalWrite(RWhPWMPin, 0);
```

```
}

void loop() {
  // Small delay for stability
  delay(100);

  // If the last command was not stop, continue at the same speed
  if (!stop) {
    analogWrite(RWhPWMPin, RSPD);
    analogWrite(LWhPWMPin, LSPD);
  }

  // Check if a new IR command is available
  if (IrReceiver.decode()) {
    int current = IrReceiver.decodedIRData.command;

    if (current == FORWARD_COMMAND) {
      // Move forward: Set forward pins to HIGH, backward pins to LOW
      digitalWrite(LWhFwdPin, HIGH);
      digitalWrite(RWhFwdPin, HIGH);
      digitalWrite(LWhBwdPin, LOW);
      digitalWrite(RWhBwdPin, LOW);
      stop = false;
    } else if (current == BACKWARD_COMMAND) {
      // Move backward: Set backward pins to HIGH, forward pins to LOW
      digitalWrite(LWhFwdPin, LOW);
      digitalWrite(RWhFwdPin, LOW);
      digitalWrite(LWhBwdPin, HIGH);
      digitalWrite(RWhBwdPin, HIGH);
      stop = false;
    } else if (current == STOP_COMMAND) {
      // Stop motors: Set speed to 0 and update stop flag
      analogWrite(RWhPWMPin, 0);
      analogWrite(LWhPWMPin, 0);
      stop = true;
    }

    // Resume IR receiver to process the next command
    IrReceiver.resume();
  }
}
```

Figure 13 is our code we used for the demo video [2]. This Arduino program controls a two-wheeled robot using an infrared (IR) remote. It employs the IRremote library to interpret signals received from an IR sensor connected to pin 7 on the Arduino. The robot can execute three commands based on signals from the remote: moving forward, moving backward, or

stopping. The motors are controlled through a motor driver module, with specific pins designated for setting motor speed (via PWM) and direction (forward or backward). Default motor speeds are set to a PWM value of 120 for both wheels. The program maintains a "stop" flag to track whether the robot is stationary or in motion. During the setup phase, the motor driver pins are configured as outputs, and the IR receiver is initialized to listen for commands. Initially, all motors are turned off. In the loop function, the program continuously monitors for IR signals. If no new command is received, the robot maintains its current speed and direction. When a new command is detected, the robot reacts by adjusting the motor control pins: enabling forward movement, backward movement, or stopping the motors entirely by setting their speed to zero. After each command is processed, the IR receiver is reset to await further input. This setup enables the robot to be remotely controlled, allowing for basic navigation and movement.

Solution Process: *Crane*

3D Modeling & Printing

The 3D modeling and printing process followed the same path as last semester. We stuck with Tinkercad to create the STL files and make Duplo-compatible component blocks. This process was streamlined due to our experience in Tinkercad from last semester. The STL files were then converted to GCODE files to be printed from the Bamboo 3D printer. Standard black PLA filament was used, the same as before.

Crane Base plate

Figure 14: Crane base plate

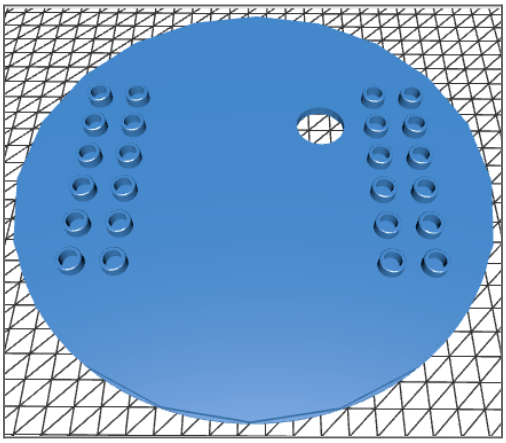


Figure 14 features the base plate that spins with the whole crane on top of it. It has duplo connections on top to attach the crane base and supports and a mount on the bottom for the servo motor underneath.

Crane Base Support

Figure 15: Crane Base Support

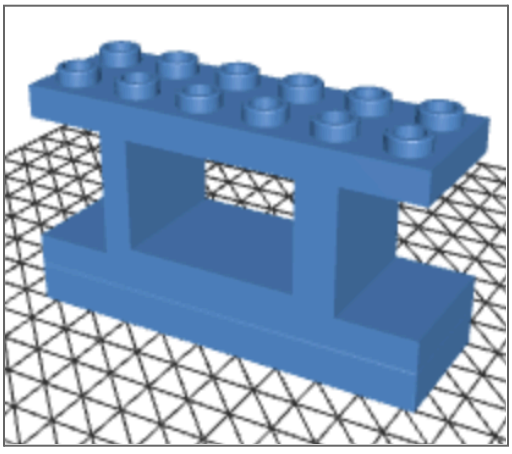


Figure 15 shows the supports that connect to the base plate and also attach via duplo to the crane base and hold the two together. It also gives room for the arduino and other components to rest on top of the base plate.

Battery Enclosure

Figure 16: Battery Enclosure

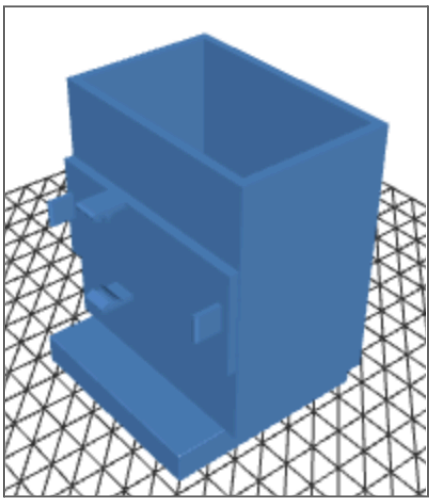


Figure 16 features an enclosure for a 12 V battery pack with a mount for a buck module. The design has a 2x4 duplo block attached to the enclosure with an open end for the battery wires to exit. The side mount holds the buck module in place, allowing the buck module contract with the batteries.

Crane Base

Figure 17: Crane Base

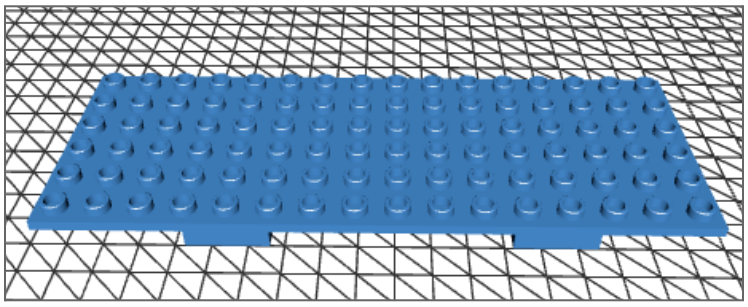


Figure 17 features the base where most of the parts are attached on top and this piece gives room for creative ideas to change around what pieces are on top of the base.

Crane Arm Base

Figure 18: Crane Arm Base

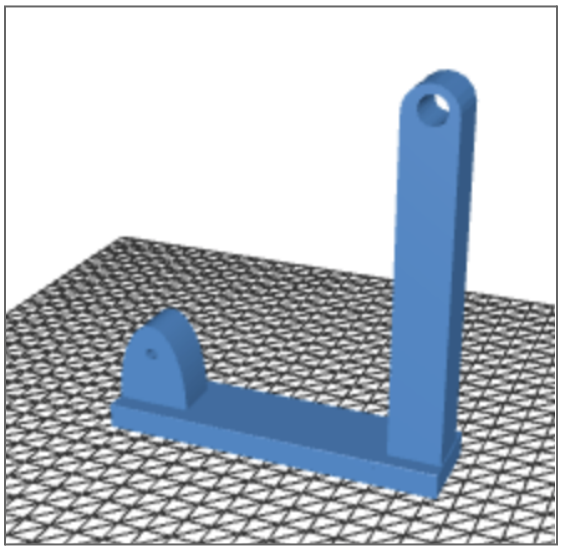


Figure 18 shows off the crane arm base which is connected by 1 2 x 8 duplo piece on the bottom to ensure the arm maintains proper distances and is out far enough over the base.

Spool Mount

Figure 19: Spool Mount

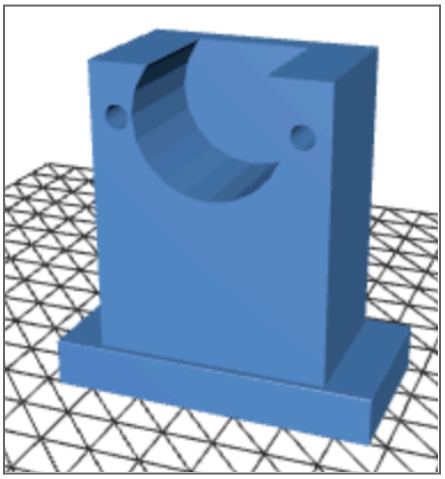


Figure 19 is a mount for the spool and servo motor. This piece rests on a 2 X 3 duplo block and sits behind the back of the crane arm base to be able to spool and pull the string back or let the string out.

Hardware/Wiring

Table 2: Crane Pin Plan

Pin	**Component**
Digital 8	*Crane Servo Motor Driver IN1*
Digital 9	*Crane Servo Motor Driver IN2*
Digital 10	*Crane Servo Motor Driver IN3*
Digital 11	*Crane Servo Motor Driver IN4*
Digital 4	*Spool Servo Motor Driver IN1*
Digital 5	*Spool Servo Motor Driver IN2*
Digital 6	*Spool Servo Motor Driver IN3*
Digital 7	*Spool Servo Motor Driver IN4*
Digital 3	*IR Sensor*
12V Power	*Spool Servo Motor Driver*
12V Power	*Crane Servo Motor Driver*
USB (Buck module 5V)	*USB Arduino*
5V Power (Arduino)	*IR Sensor*

Component Integration

The components of the crane robot follow the same pattern as our initial robot car. We used a lot of pieces that we have used in past classes and we also used parts from our robot car. The only new electrical components we introduced were a servo motor and the servo motor

driver. The servo for the arm of the crane wasn't strong enough so we 3D printed a gearbox to improve the torque. Other than that, the motors worked perfectly.

Code

Figure 20: Code

```
#include <AccelStepper.h>
#include <IRremote.h>

#define IR_Pin 3

#define craneForward 12 //1
#define craneReverse 24 //2
#define craneStop 94 //3

#define spoolForward 8 //4
#define spoolReverse 28 //5
#define spoolStop 90 //6

#define IN5 4
#define IN6 5
#define IN7 6
#define IN8 7

#define IN1 8
#define IN2 9
#define IN3 10
#define IN4 11


bool stop = false;

AccelStepper spoolSpin(AccelStepper::HALF4WIRE, IN5, IN6, IN7, IN8);
AccelStepper craneSpin(AccelStepper::HALF4WIRE, IN1, IN3, IN2, IN4);

void setup() {
  Serial.begin(9600);  // Initialize serial communication for debugging
  IrReceiver.begin(IR_Pin, ENABLE_LED_FEEDBACK); // Assuming this is correct

  spoolSpin.setMaxSpeed(1500);  // Max speed set to 1500 steps/sec
  spoolSpin.setAcceleration(500);  // Acceleration set to 500 steps/sec²
  spoolSpin.setSpeed(0);  // Initial speed set to 1000 steps/sec

  craneSpin.setMaxSpeed(1500);  // Max speed set to 1500 steps/sec
  craneSpin.setAcceleration(500);  // Acceleration set to 500 steps/sec²
  craneSpin.setSpeed(0);  // Initial speed set to 1000 steps/sec
}

void loop() {
```

```
  craneSpin.runSpeed();  // Keep the motor running at set speed
  spoolSpin.runSpeed();  // Keep the motor running at set speed

  if (IrReceiver.decode()) {
    int current = IrReceiver.decodedIRData.command;

    // Debugging statement
    Serial.print("IR Received: ");
    Serial.println(current);  // Print the received IR command
    if (current == spoolForward) {
      spoolSpin.setSpeed(1000);  // Set speed to 1000 steps/sec
    } else if (current == spoolReverse) {
      spoolSpin.setSpeed(-1000);  // Set speed to -1000 steps/sec
    } else if (current == spoolStop) {
      spoolSpin.setSpeed(0);  // Stop the motor
    }

    if (current == craneForward) {
      craneSpin.setSpeed(1000);  // Set speed to 1000 steps/sec
    } else if (current == craneReverse) {
      craneSpin.setSpeed(-1000);  // Set speed to -1000 steps/sec
    } else if (current == craneStop) {
      craneSpin.setSpeed(0);  // Stop the motor
    }

    IrReceiver.resume();
  }
}
```

Figure 20 shows our Arduino program that controls two stepper motors. One for a crane and one for a spool using IR remote commands. It uses the AccelStepper library to manage motor speed and acceleration, and the IRremote library to decode signals from an IR receiver connected to pin 3. Each motor is controlled through specific IR codes: the crane responds to commands for forward, reverse, and stop using defined codes (12, 24, 94), and the spool responds similarly (8, 28, 90). The setup() function initializes serial communication, the IR receiver, and motor settings. In the loop(), both motors continuously run at their set speed, and any incoming IR command updates the speed and direction of the corresponding motor. This

setup enables remote, real-time directional control of two stepper motors, suitable for a basic crane system.

Final Results & Future Work

For the *Fall 2024* semester we have completed project research, 3D printing specialized duplo blocks, powering and wiring the robot, writing a simple piece of code for maneuverability, assembly of the duplo blocks in various positions, and documentation. A sample robot demo can be found here showcasing a possible robot configuration that runs the code shown in Figure 9.

Figure 21: Full Car Robot

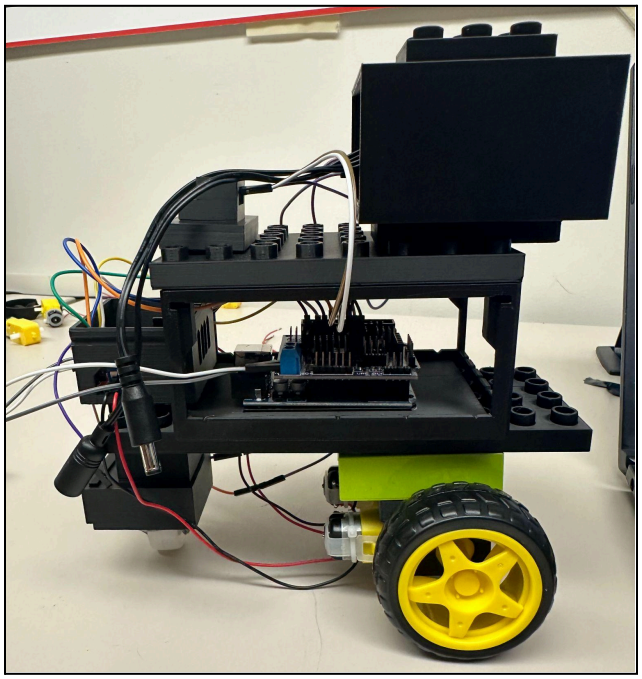


Figure 21 contains a picture of our car robot. The video of the robot [2] features the basic maneuverability of the robot. The design in Figure 21 works well with weight distribution. We placed the battery pack directly above the wheels to keep the robot balanced. Everything is placed within the wiring distance of the Arduino Uno. We can work on discrete wiring, weight reduction, and adding more components to enhance the robot in the future.

For the *Spring 2025* semester, we completed another complete robot with our designed pieces. This included research on how cranes worked and what pieces were necessary to have a functional crane and the same 3d modeling and printing route as the Fall semester.

Figure 22: Full Crane Robot

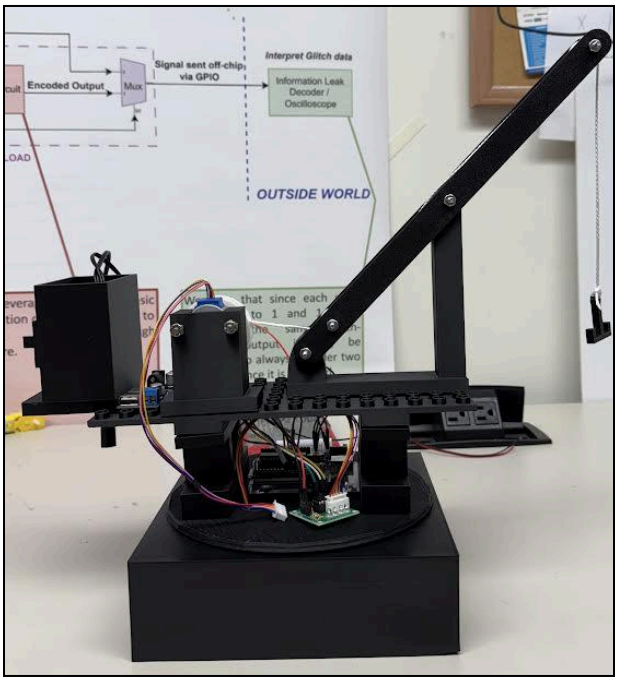


Figure 22 contains a picture of our crane robot. The design in Figure 22 can lift up to two pounds. Everything is placed within the wiring distance of the Arduino Uno. We can work on discrete wiring, weight reduction, and adding more components to enhance the robot in the future.

With the foundational goals for our project successfully achieved, our focus now shifts to expanding the capabilities of the Duplo-based robot. Future work will involve:

Adding More Sensors:

- Touch Sensors: These sensors will provide tactile feedback, enabling the car robot to respond to physical interactions, such as stopping when an obstacle is touched.
- Headlights: LED modules integrated into Duplo blocks to improve visibility in low-light environments and signal different states.
- Color Detectors: RGB sensors can enhance the robot's ability to differentiate between colors, opening possibilities for tasks like color-coded sorting or following colored paths.

Discrete Wiring:

- Developing a more efficient way of wiring the sensors to decrease the minimum age requirement for the product.
- Possibly use a rail system for power and signal transfer instead of wires.

Proof of Concept

Figure 23: Alternate Car Design Front View

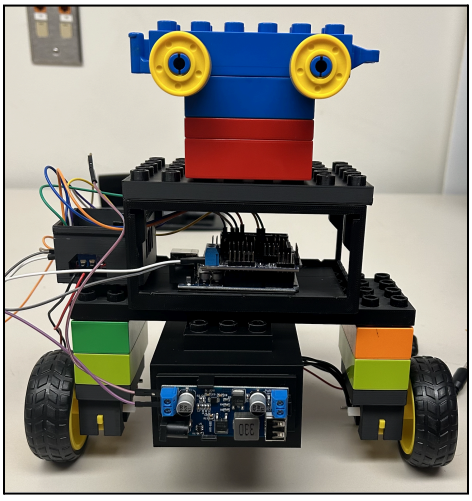
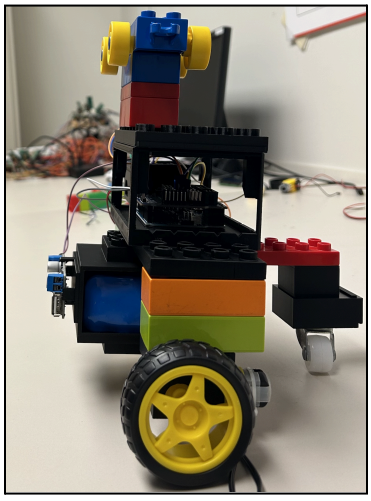


Figure 24: Alternate Car Design Side View



In Figures 23 and 24 we developed a different robot with Legos' duplo blocks. The design resembles a crab-like toy to appeal to younger generations. We can work on fixing the wiring to be more discrete and easy to move.

Conclusion

While working through our senior design project, we have gone through the main stages of the design process. Starting with an idea, we had to first create goals and a vision of what we wanted to accomplish. Our main goal was to prototype a Duplo version of the Lego Mindstorm including individual components that can be configured however the user wants. After the initial planning stage, we had to research methods of 3D printing and components that would be easily integrated into a circuit. From there we went straight into the design which included multiple designs that were made specifically for each component that we wanted to include with the final product. We then printed these components with the Bambu P1S. After each component was created, we went into the testing/building stage which included integrating our 3D-printed parts with our electrical components. This led us to our final stage of analysis where we fully documented all of our work into a final paper. The design process can be time- consuming, but with this knowledge we can go into the work field with a newfound experience under our belts.

Reflection

This senior design project deepened our understanding of professional and ethical responsibilities as engineers. We kept creativity and freedom at the heart of our project because that's what we believe our project was all about. Developing a tool aimed at helping children explore robotics proved exceptionally rewarding, allowing us to see firsthand how engineering solutions can influence education positively.

1. How does your solution affect or may potentially affect public health, safety, and welfare? What are your considerations and reflections on global, cultural, social, environmental, and economic factors in your solution?

Our Duplo-based robotic system supports cognitive and problem-solving skills in children, positively impacting educational outcomes and social well-being. Safety was paramount in our design process, reducing potential hazards through careful consideration of materials and construction methods. On a global scale, our project provides a universally accessible introduction to robotics, appealing to diverse backgrounds and economic situations. Although environmental concerns were raised due to our use of 3D-printed plastics, the reusable and modular nature of our system helps mitigate waste. Economically, our commitment to affordability through accessible materials and open-source technologies ensures broad applicability in educational settings worldwide.

2. Discuss your team's professional and ethical responsibility as engineers and the impact that your solutions may have in the global, economic, environmental, and societal context.

Our role as engineers compelled us to create safe, inclusive, and socially beneficial designs. Ethical responsibility guided our decisions, particularly regarding safety and educational value. By creating an accessible educational tool, we aim to reduce barriers in STEM education globally, offering a cost-effective alternative for families and underfunded educational institutions. Environmentally, despite the plastic use inherent in 3D printing, our sustainable design approach prioritizes longevity and adaptability. Ultimately, our project promotes curiosity, hands-on learning, and inclusivity, positively shaping future generations.

3. How you acquired and applied new knowledge as needed

Throughout the project, we consistently gained new knowledge in areas such as microcontroller programming, 3D modeling with Tinkercad, and practical 3D printing techniques. We learned how to integrate mechanical components effectively, addressing challenges related to torque and wiring complexities through iterative prototyping. Leveraging real-world problem-solving approaches allowed us to refine our designs continuously, improving system performance and usability. Adopting this proactive learning mindset greatly enhanced both our technical expertise and our collaborative skills as a cohesive engineering team.

Works Cited

[1]

"Dana clark/DuploMindstorm: ECE 448/449 Senior Capstone Project." *GitHub*, github.com/danaeclark/DuploMindstorm. Accessed 4 Dec. 2024.

[2]

"ECE 448: Duplo Mindstorm, Fall 2024 Final Result." *YouTube*, YouTube, www.youtube.com/watch?v=XVqvp0UVz1A. Accessed 4 Dec. 2024.

[3]

"Doblo Factory." *Doblo Factory - EduTech Wiki*, edutechwiki.unige.ch/en/Doblo_factory. Accessed 12 Nov. 2024.

[4]

"Lego Duplo Brick Collection with Parametric Source Files." *Lego Duplo Brick Collection with Parametric Source Files by András Bognár | Download Free STL Model | Printables.Com*, www.printables.com/model/340295-lego-duplo-brick-collection-with-parametric-source. Accessed 12 Nov. 2024.

[5]

"Lego Duplo Compatible Building Block 2x2." *Lego Duplo Compatible Building Block 2x2 by TmnSn | Download Free STL Model | Printables.Com*, www.printables.com/model/137952-lego-duplo-compatible-building-block-2x2. Accessed 12 Nov. 2024.

[6]

"Search for 3D Designs and Circuits." *Tinkercad*, www.tinkercad.com/search?q=Duplo+block&staffPicks=0. Accessed 12 Nov. 2024.

[7]

"Subject & Course Guides: Making from Home Resources: 3D Modeling." *3D Modeling - Making From Home Resources - Subject & Course Guides at Miami University*, libguides.lib.miamioh.edu/make-from-home/tutorials-3D-modeling. Accessed 12 Nov. 2024.