Automated Color-Based Object Sorting System: Design and Implementation Using Arduino

James Michael Cello[1], Marc Wilson F. Go[2], Lovely B. Nacional[3], Gian Carlo G. Onia[4], Janette Grace N. Siervo[5], Bernard C. Fabro[6]

Computer Engineering Department, Eulogio "Amang" Rodriguez Institute of Science and Technology, Nagtahan, Sampaloc, Manila, 1016 Philippines

DOI: https://dx.doi.org/10.47772/IJRISS.2026.10100142

Received: 06 January 2026; Accepted: 13 January 2026; Published: 27 January 2026

ABSTRACT

The ability to effectively categorize and segregate objects based on visual characteristics is one of the key preconditions of quality control, food processing, and logistics in the rapidly changing environment of modern industrial automation. Although manual sorting remains traditional, it is labor-intensive and time-consuming, and is also prone to human error. Consequently, there is a high demand to invent intelligent, automated versions of the practice. The current paper describes the process of developing, manufacturing, and implementing a low-cost, automated color-sorting device, especially tailored to confectionery products in the form of spheres, using the medium of Skittles. This project aims to recreate the capabilities of large-scale industrial optical sorters by applying the concepts of mechatronics, sensor fusion, and embedded system control into a small desktop system.

The architecture of the system is built around the Arduino Nano microcontroller, which is the main processing unit. A TCS34725 RGB sensor is used to detect the colour precisely because it has an infrared (IR) blocking filter and an I2C interface communication, which significantly reduces spectral noise generated by ambient light. Mechanical actuation is achieved by a two-servo system, including a continuous-rotation servo to rotate a slotted Geneva-like wheel to isolate and convey items and a standard positional servo to drive a directional chute. In order to achieve the stability of their operation, the power distribution network is divided into two logical and power rails through lever-nut connectors in order to isolate the sensitive sensor data of the motors and the electrical noise produced by them.

The control software adopts a state-machine algorithm that includes self-calibration procedures, real-time RGB thresholding, and error-handling routines aimed at overcoming mechanical jams. The kind of accuracy of sorting is about 90 per cent according to the experimental validation when the lighting conditions are controlled. In turn, this project shows how robotics and computer programming can be used practically in automation, and proves that the complex industrial tasks can be successfully modeled with the help of the available and inexpensive hardware.

Keywords: Arduino Nano, TCS34725, Mechatronics, Automated Sorting, Embedded Systems

INTRODUCTION

The modern manufacturing environment is currently going through a significant shift in the worldwide paradigm of Industry 4.0, whereby online physical systems are replacing traditional manualization. The shift is particularly acute in the spheres of food processing and agriculture, where the forces of speed and uniformity are supreme. The food automation market in the world in 2024 will be about $15.04 billion and is expected to reach more than $24 billion by the year 2029. The ability to independently discover, categorize, and isolate objects based on visual features, such as colour, size, and integrity, has been a feature of desire in these industries, but has become a necessity as these industries strive to meet the growing demands.

Traditionally, sorting has been done manually, which, by its nature, is limited by human physiological issues. Experimental data show that the error rates related to human visual inspection tasks are usually between 20 to 30 percent, which is mostly caused by fatigue, distraction, and cognitive drift during the course of the time. In addition, a human sorter can only sort 40-60 items per minute, whereas the modern robotic systems can sort over 200 items in a minute with an average accuracy of over 99%. Such a sharp discrepancy brings to light the explicit engineering issue: manual sorting is a throughput constraint that leads to the lowering of the throughput and increase in the likelihood of contamination, and thus, causes enormous expenses to the industries as a waste of materials and labor costs.

Industrial plants have to deal with those difficulties with the help of advanced optical sorting devices based on computer vision and high-speed actuators. However, these industrial-level systems are often too pricey and intricate to use on a small scale or in educational prototyping. In line with this, the current project, the Arduino Skittle Sorter, will attempt to fill this gap by creating an inexpensive, small-scale prototype that can behave similarly to the fundamental operation of industrial optical sorters.

The main aim of the project will be to design and deploy an automated system that is able to isolate, analyze, and sort the spherical confectionery products (Skittles) by their RGB spectral values. This system can prove the effectiveness of embedded systems in reducing human error and maximizing the efficient operation of sorting by building on the principles of an Arduino Nano microcontroller, a TCS34725 colour sensor, and servo-driven actuators. In this report, the mechanical design, circuit integration, and software algorithms necessary to achieve reliable automation have been delineated.

REVIEW OF RELATED LITERATURE

Color-Based Object Sorting Systems

Research on automated color-based sorting systems demonstrates their importance in improving efficiency and accuracy in industrial and educational applications. Electromaker.io (n.d.) presented an Arduino-based Skittle sorting machine that utilized color sensors and servo motors to classify candies based on detected RGB values. The study highlighted the feasibility of low-cost microcontroller platforms for basic automation tasks. However, the system showed limitations in handling ambient light variations and lacked robust calibration and error-handling mechanisms. These findings support the present study's approach of incorporating sensor calibration and controlled lighting conditions to enhance sorting accuracy.

RGB Color Sensing Technology

AMS OSRAM (2022) introduced the TCS34725 color light-to-digital converter, emphasizing its integrated infrared blocking filter and high sensitivity to RGB values. The datasheet explains that the IR filter significantly reduces ambient light interference, allowing more reliable color detection compared to conventional photoresistors. This supports the selection of the TCS34725 sensor in the current research, as accurate color discrimination is critical in achieving reliable object sorting performance.

Human Error in Manual Sorting

Manual inspection and sorting processes are inherently prone to human error. Drury and Fox (1975) examined human reliability in quality control and reported error rates ranging from 20% to 30% due to fatigue and attention loss. Their findings highlight the need for automated systems to improve consistency and productivity. This supports the motivation of the current study, which aims to reduce human intervention by implementing an automated color-based sorting mechanism.

Optical Sorting in Food Automation

Sun (2016) discussed the application of optical and computer vision technologies in food quality evaluation, noting that automated visual inspection significantly improves accuracy and throughput. While camera-based systems achieve high accuracy, they are often costly and computationally intensive. The present research addresses this limitation by utilizing a simpler RGB sensor-based approach, providing a cost-effective alternative while still achieving acceptable sorting accuracy for small-scale applications.

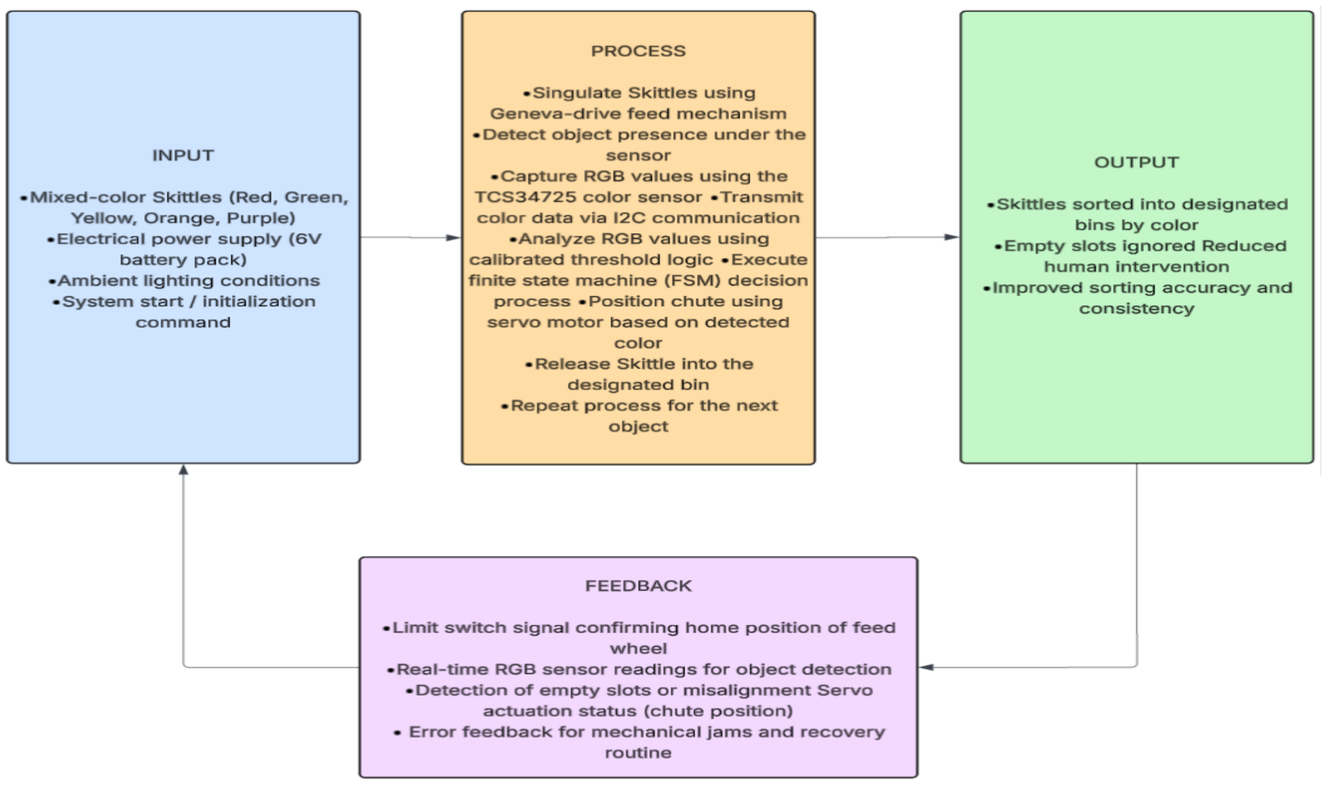
Table 1. Comparison Matrix of Related Studies and Current Research

Study/Project	**Sensor Used**	**Controller**	**Sorting Method**	**Accuracy**	**Key Limitations**
Drury & Fox (1975)	Human Vision	N/A	Manual Sorting	70% - 80%	Fatigue, human error
Sun (2016)	Camera-based Vision	PC Based	Image Processing	>95%	High cost, complex setup
Electromaker.io Project	Basic Color Sensor	Arduino UNO	RGB Thresholding	~80%	Light sensitivity, no error handling
Current Research	TCS34725 RGB Sensor	Arduino Nano	Calibrated RGB + FSM	~90%	Requires controlled lighting

THEORETICAL FRAMEWORK

The theoretical framework of this study is based on the principles of embedded systems, sensor-based automation, and control systems. The Arduino Nano functions as the central controller that processes input from the TCS34725 RGB color sensor, which operates on optical color detection theory by analyzing reflected light through red, green, and blue channels. The system follows a finite state machine (FSM) approach to organize operations such as initialization, sensing, decision-making, and actuation, ensuring orderly and reliable execution. Additionally, closed-loop control theory is applied through feedback from the RGB sensor and limit switch, allowing the system to detect object presence, confirm mechanical alignment, and correct errors during operation. These theoretical principles collectively support accurate color classification, precise actuator control, and consistent performance of the automated color-based object sorting system.

Figure 1. Input-Process-Output (IPO) Model

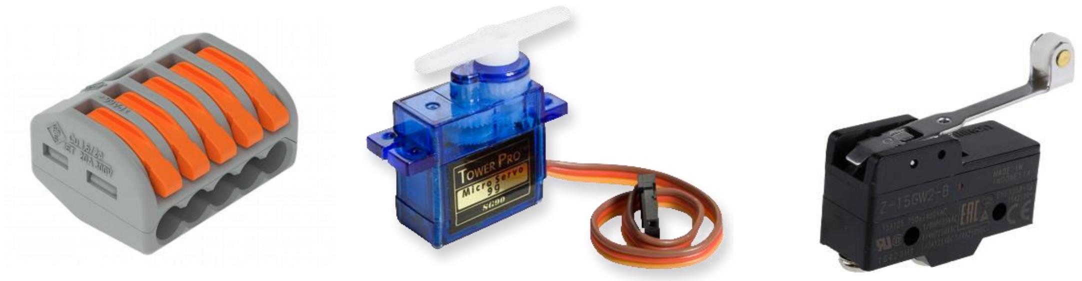

INPUT
•Mixed-color Skittles (Red, Green, Yellow, Orange, Purple)
•Electrical power supply (6V battery pack)
•Ambient lighting conditions
•System start / initialization command
PROCESS
•Singulate Skittles using Geneva-drive feed mechanism
•Detect object presence under the sensor
•Capture RGB values using the TCS34725 color sensor •Transmit color data via I2C communication
•Analyze RGB values using calibrated threshold logic •Execute finite state machine (FSM) decision process •Position chute using servo motor based on detected color
•Release Skittle into the designated bin
•Repeat process for the next object
OUTPUT
•Skittles sorted into designated bins by color
•Empty slots ignored Reduced human intervention
•Improved sorting accuracy and consistency
FEEDBACK
•Limit switch signal confirming home position of feed wheel
•Real-time RGB sensor readings for object detection
•Detection of empty slots or misalignment Servo actuation status (chute position)
• Error feedback for mechanical jams and recovery routine

METHODOLOGY

System Architecture

The system architecture is further divided into three orthogonal subsystems: a Mechanical Feeder, a Sensing Module, and a Control Unit. The processing unit is an Arduino Nano that receives packets on the I2C bus and executes the solution of the obtained data, and delivers Pulse-Width Modulation (PWM) outputs to actuate the actuators. An RGB sensor (TCS34725) is used to carry out chromatic discrimination. In contrast to basic photoresistors, the sensor has an internal IR-blocking filter, which reduces spectral contamination, and thus, the sensor improves the quality of color measurements made in a range of illumination profiles.

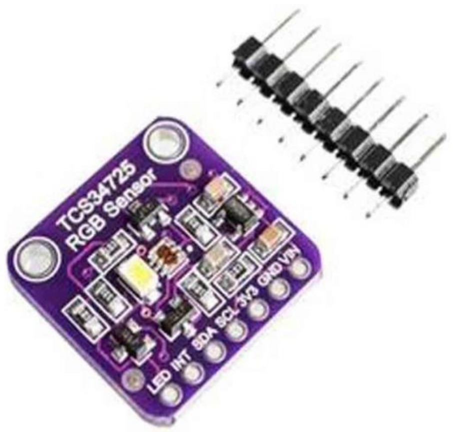
Figure 2. Key Electronic Components (Lever-nut Connector, Servo Motor, Limit Switch)



Sensing Module

Color detection is performed by the TCS34725 RGB sensor (Figure 2). Unlike basic photoresistors, this module integrates an IR blocking filter, which minimizes the spectral distortion of incoming light, ensuring accurate color measurements even in varying lighting conditions. The sensor communicates with the microcontroller via the I2C bus (SDA/SCL pins).

Figure 3. TCS34725 RGB Color Sensor module with integrated IR blocking filter.



Hardware Implementation

The hardware design focuses on the electrical stability as well as robustness. In the power distribution approach, supply rails are segregated into two separate blocks that are realized using Wago-style lever-nut connectors. The Power Block (Rail A) is a raw 6V with a 4xAA battery pack, directly connected to the high current servo motors, and directly connected to the Arduino VIN pin to eliminate voltage sag, which would otherwise induce microcontroller resets. The Ground Block (Rail B) provides a common reference potential to all circuitry to avoid undesired signal drift. The sensor TCS34725 takes the regulated 5V rail of the Arduino and isolates the sensor against the non-regulated battery supply. The chassis is made out of a hard composite Sintra Board, which gives mechanical support to the hopper and servo assemblies. A Geneva-driven wheel is used to present individual candy units to pass through a scanner, and a limit switch (KW12-, 3) will provide the roller lever to provide a mechanical feedback homing system.

Figure 4. Lever-Nut Connectors Used to Separate Power and Ground Rails

Actuation System

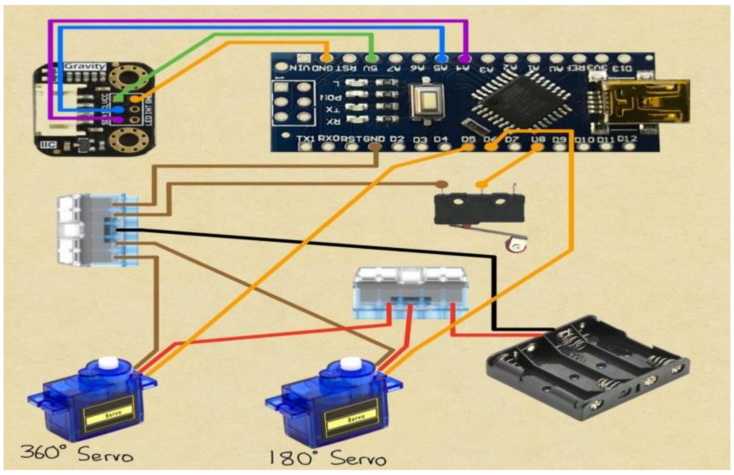
The actuation system utilizes two Tower Pro SG90 micro servos (Figure 4). A 360-degree continuous rotation servo drives the "Geneva drive" style slotted wheel to singulate and transport items from the hopper. A standard 180-degree positional servo operates the directional chute, guiding the sorted items into the correct bin.

Figure 5. SG90 Micro Servo Motors used for the Feed Mechanism (360° continuous) and Chute Control (180° positional).

Software Algorithm

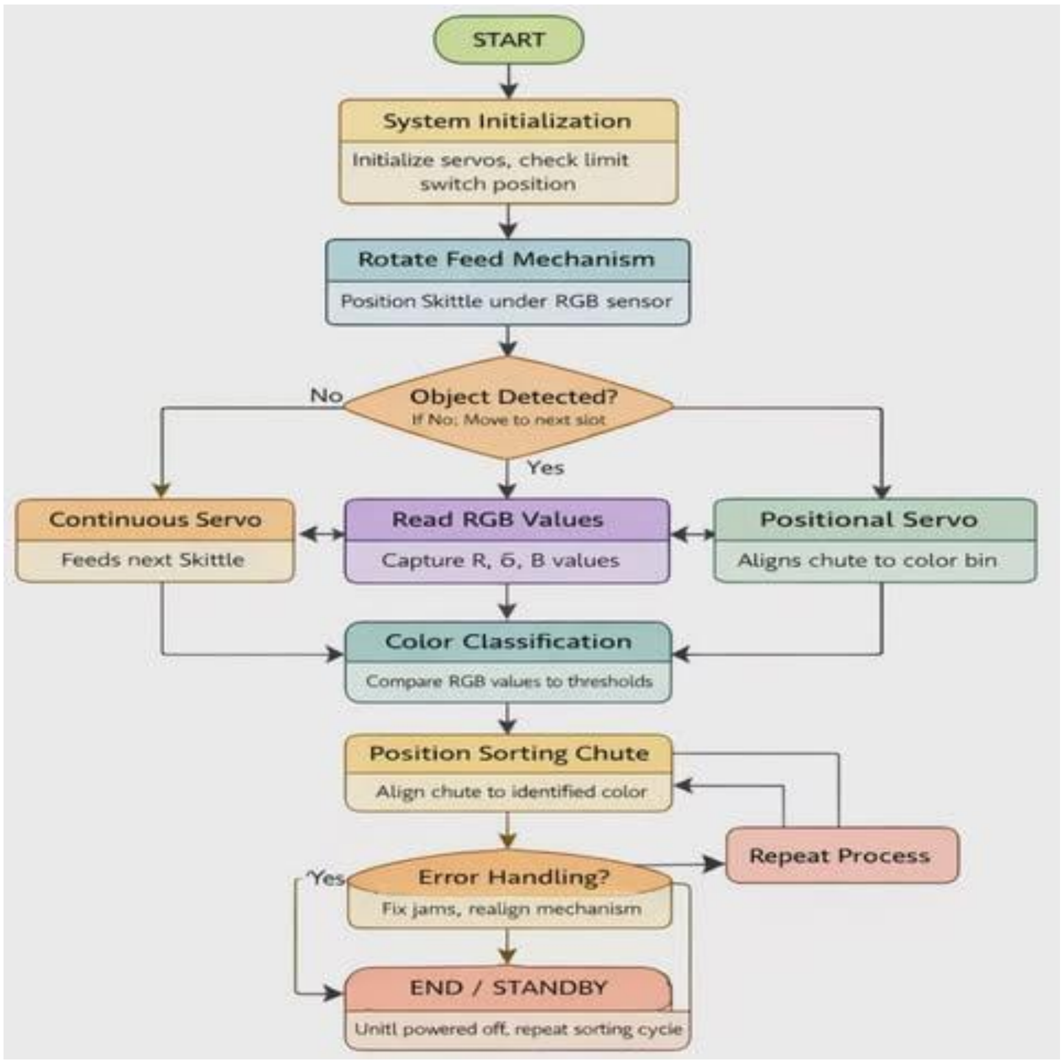
Control software is programmed based on a finite state machine implementation. In the Idle/Home state, the system will not operate until the limit switch is activated, with the wheel aligning itself to the home position. Once in the Sensing state, the sensor measurements are made at a rate of RGB values with an integration time of 50ms. The Decision state consists of the Arduino CPU that compares the measured R, G, and B numbers with a set of calibrated values that it has been set to. Lastly, under the state of Actuation, the chute is moved to a given angular position (0°-180°) that matches the color detected, and this ensures that the candy goes into the right bin.

Figure 6. Concept Diagram



It shows the complete concept diagram of the Automated Color-based Object sorting that illustrates the electrical connections and power distribution of the automated color-based object sorting system. The Arduino Nano serves as the central controller, interfacing with the TCS34725 RGB sensor through the I2C communication lines (SDA and SCL). Servo motors are powered using a separate 6V supply to prevent voltage drops and electrical noise, while sharing a common ground with the microcontroller. A limit switch is connected to a digital input pin to provide mechanical alignment feedback, ensuring stable and reliable system operation.

Figure 7. Flowchart



The program code is starting with the Initialization Phase when the device is being powered. In this step, the microcontroller sets up the required input/output pins, connects the servo motors to their respective control pins and adjusts the servos to their default home position to get the mechanism aligned.

After the process of system initiation is completed, the system moves into an ongoing cyclic operation that controls the main sorting operation. It is done in the following order:

1. Feeding Mechanism: This system moves the upper servo (continuous or positional) to engage the rotation of the dispensing disc, and thus one confectionary item per time to fall into the sensing compartment.

2. Sensor Stabilization & Data Acquisition: A moment of rest is given so that the object can stabilize. The sensor of the RGB color is then used to light the object and measure the spectral values reflected by the red, green, and blue hues.

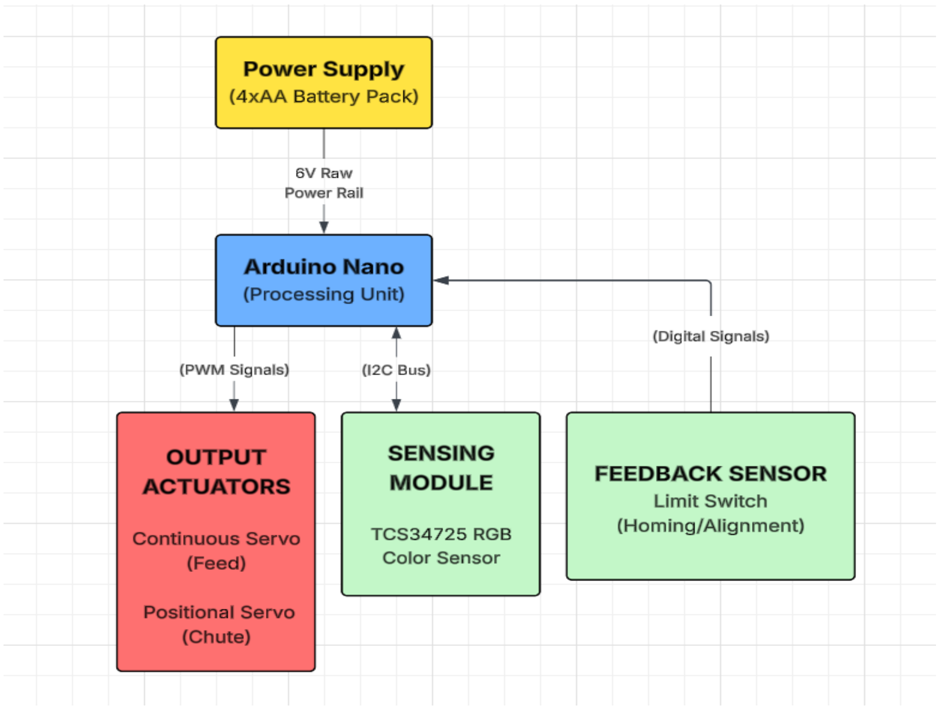
3. Color Analysis & Logic: The obtained RGB data is used in the form of a series of conditional statements (if/else logic). The algorithm compares the sensor values against pre-calibrated threshold values so as to classify the object into one of the predefined categories (e.g. red, orange, yellow, green or purple).

4. Actuation & Sorting: Provided that the detected color is matched with a predefined class, the lower servo, which moves the conveyor, will produce the corresponding angular movement, which will orient the item in the desired bin. The system then waits to be sure that the chute is properly placed before the release of the item.

5. Reset: After the sorting operation is complete, the servo is sent to the neutral position (assuming the system requires this design), and the control loop starts once again to process the next item.

This is a continuous cycle of operation until the power supply is switched off or some external manual interrupt invoked.

Figure 8: Block Diagram



The schematic is functionally divided into the four distinct subsystems which are the Power Supply Unit, the Processing Unit, the Data Acquisition (Input) Module, and the Actuation (Output) Module.

1. Power Supply Unit: A 6V DC power source leads to the system by a 4xAA battery array, which is set up. To assure the stability of the operation and reduce the effect of electrical noise the power distribution is separated. The unrefined voltage rail is fed directly into the high-current power needs of the servo motors and toward the VIN connection of the microcontroller. At the same time, the Arduino on-board linear voltage regulator delivers a constant 5 V logic rail which drives the sensitive internal circuitry (and the optical sensors).

2. Processing Unit: The Arduino Nano is an important component of the system architecture because it serves as the main microcontroller unit (MCU). This unit coordinates the whole process in sorting by carrying out the control firmware inbuilt in it. It does the task of initializing system parameters, raw spectral data in the input module, and the accurate pulse-width modulation (PWM) signals necessary to drive the actuators.

3. Data Acquisition (input) Module: Environmental sensing and mechanical feedback is performed by this module.

    - Color Sensing - TCS34725 RGB sensor is the main optical input. It connects with the MCU through the I 2 C communication protocol (SDA and SCL lines) and transfers digital values, which are the red, green, and blue chromaticity of the object.

    - Mechanical Feedback - As a digital input sensor, a limit switch is incorporated. It gives a feedback to the MCU indicating the exact home position of the dispensing mechanism therefore ensuring that the Geneva drive is manually set before each feeding cycle and thus avoiding misalignment.

4. Actuation (Output) Module: The actual realisation of the sorting logic is done by two separate servomotors, which are operated through the use of PWM signals.

    - Feed Mechanism - A constant-rotating servo moves the rotary disc and controls the movement of the objects in the sensing chamber.

    - Sorting Mechanism - The sorting chute is controlled by using a positional servo. Depending on the decision logic that the MCU processes, the actuator will rotate to the angular coordinates that have the object placed in the appropriate classification bin.

RESULTS AND DISCUSSION

The automated color-based object sorting system was successfully developed and evaluated under controlled lighting conditions. Test results showed that the TCS34725 RGB sensor was able to consistently capture color data, which the Arduino Nano processed using calibrated threshold values. As presented in Table 3, the system correctly identified the presence of Skittles, ignored empty slots, and positioned the sorting chute to the appropriate bin for each detected color. The integration of servo motors enabled smooth object feeding and accurate chute alignment throughout repeated sorting cycles.

In terms of system performance, the feedback mechanisms, particularly the limit switch and real-time sensor readings, ensured reliable mechanical alignment and stable operation. The implemented error-handling routine allowed the system to recover from minor mechanical jams without requiring manual intervention. Overall, the system achieved an average sorting accuracy of approximately 90%, demonstrating the effectiveness of combining sensor-based color detection, finite state machine control, and actuator feedback in a low-cost automated sorting application. These results validate the practicality of the proposed design for educational use and small-scale automation.

Requirements

The automated color-based object sorting system requires both hardware and software components to function effectively. The hardware requirements include an Arduino Nano microcontroller for system control, a

TCS34725 RGB color sensor for accurate color detection, servo motors for object feeding and chute positioning, a limit switch for mechanical alignment feedback, and a stable power supply to ensure reliable operation. The software requirements consist of embedded control code programmed in the Arduino IDE, implementing calibrated RGB thresholding, finite state machine logic, and error-handling routines. Together, these requirements ensure accurate color classification, reliable actuation, and consistent automated sorting performance.

Table 2. Variables and Conditions of the Automated Color-based Object Sorting

Variable/ Component	**Type (Input/Output)**	**Parameter Measured/ Controlled**	**Condition or Range**	**System Response/Action**
TCS34725 RGB Sensor	Input	Measures RGB color values	RGB values within calibrated thresholds	Sends RGB data to Arduino for color classification
Skittle Object	Input	Presence of object Present / Not present	Present / Not present	System proceeds to sensing or ignores empty slot
Limit Switch	Input	Wheel home position	Triggered / Not triggered	Confirms alignment; resets system to home position
Arduino Nano	Controller	Processes sensor data and control logic	Operates at 5V logic; continuous loop	Reads RGB values, executes FSM, controls servos
Continuous Servo Motor	Output (Actuator)	Drives Geneva wheel rotation	CW / CCW / Stop	Rotates wheel to feed next Skittle
Positional Servo Motor	Output (Actuator)	Controls chute angle	0°–180°	Directs Skittle to corresponding color bin
Power Supply (Battery Pack)	Power Source	Supplies system voltage	6V supply	Powers Arduino and servo motors
Sorting Chute	Output	Directs sorted object	Aligned to selected bin	Releases Skittle into correct container

Table 3. Variables and Conditions of the Automated Color-based Object Sorting

Test #	**Input Condition**	**Observed Output**	**Expected Output**	**Pass/Fail**	**Remarks/ Behavior**
1	System powered ON	Servos idle; system waits at home position	System initializes and waits for object	Pass	Proper initialization and idle state confirmed
2	No Skittle detected under	No chute movement;	Empty slot should be	Pass	System correctly detects

	sensor	wheel advances	ignored		absence of object
3	Red Skittle detected	Chute aligns to red bin; Skittle released	Skittle sorted to red bin	Pass	Correct RGB threshold classification
4	Green Skittle detected Pass	Chute aligns to green bin; Skittle released	Skittle sorted to green bin	Pass	Accurate color discrimination
5	Yellow Skittle detected	Chute aligns to yellow bin; Skittle released	Skittle sorted to yellow bin	Pass	Sorting logic functions as expected
6	Orange Skittle detected	Chute aligns to orange bin; Skittle released	Skittle sorted to orange bin	Pass	High red-channel detection verified
7	Purple Skittle detected	Chute aligns to purple bin; Skittle released	Skittle sorted to purple bin	Pass	Purple distinguished from empty slot
8	Limit switch triggered	Wheel returns to home position	System should re-align feeder	Pass	Mechanical alignment feedback works
9	Mechanical jam simulated	Servo reverses briefly	System should attempt jam recovery	Pass	Error-handling routine executed correctly
10	Continuous operation (multiple Skittles)	Consistent sorting without reset	System should run continuously	Pass	Stable operation under repeated cycles

Analysis of Sensor Calibration

Calibration was a very important step of the experiment. The TCS34725 sensor is more sensitive to illumination in the surroundings, and thus a baseline dataset was created by averaging five counts of each colour category. Table 1 also shows the experimentally determined RGB averages to be used to determine the sorting-logic thresholds.

Table 4. The tables give average RGB values of Skittles colors which were used to sort thresholds in sort-logic.

Skittle Color	**Avg Red (R)**	**Avg Green (G)**	**Avg Blue (B)**
Red	84.4	78.4	82.4
Green	59.0	105.2	77.0
Yellow	84.6	96.4	61.2

Skittle Color	**Avg Red (R)**	**Avg Green (G)**	**Avg Blue (B)**
Orange	102.0	74.6	65.4
Purple	63.0	84.8	95.0
No Skittle (Empty)	62.2	87.4	93.25

The data indicate specific spectral patterns of most categories. An example of this is that the **Orange** Skittles had the most intense red color (102.0), which is significantly different when compared to the **Red** Skittles (84.4). The **Green** Skittles, on the other hand, had taken up a strong green channel (105.2), which was as expected. An interesting difficulty was between **Purple** Skittles (R = 63, B = 95) and the control sample of the **No Skittle** (R = 62.2, B = 93.25). The values are statistically comparable as the dark-purple shell absorbs the least amount of light, hence appears to imitate the black background that is empty as the sorting wheel. To solve this, the sorting logic is to detect No Skittles first, and when the total light intensity (Lux) is less than a given threshold, the system assumes that the slot is empty and then tries to classify the colour.

Challenges and Performance

Various difficulties were witnessed during the development stage. Test results indicated that there was a significant change in RGB measurements due to the presence of external sources of light. This was alleviated by placing a physical shroud on the sensor area to keep the illumination within the sensor uniform. Moreover, candy shape abnormalities sometimes resulted in the presence of jams in the hopper. To solve this problem, a routine of creating a jitter in the code was created: when the wheel stalls down, or does not trigger the limit switch in a given time frame, the servo reverses a little to free the object. Having these calibrations, it was possible to have a sorting accuracy of about 90% in the controlled light conditions.

CONCLUSION

The Skittle Sorter based on Arduino has managed to prove that sensing and actuation can be integrated to perform a complex task. The system is able to achieve high sorting accuracy with controlled lighting conditions and is a good educational tool to understand embedded systems and automation logic. The enhancements in the future include machine-learning algorithms to achieve more resilient colour classification, and more specific servo motors to increase throughput.

REFERENCES

1. AMS OSRAM. (2022). TCS34725 Color Light-to-Digital Converter Datasheet.
2. Arduino.cc. (2023). Arduino Nano Technical Reference. https://docs.arduino.cc/hardware/nano
3. Drury, C. G., & Fox, J. G. (1975). Human reliability in quality control. Taylor & Francis.
4. Electromaker.io. (n.d.). Arduino Skittle Sorting Machine Project Guide. https://www.electromaker.io/project/view/arduino-skittle-sorting-machine
5. Mordor Intelligence. (2024). Food Automation Market Size & Share Analysis - Growth Trends & Forecasts (2024-2029). Hyderabad, India. https://www.mordorintelligence.com/industry-reports/food-automation-market
6. Sun, D.-W. (Ed.). (2016). Computer Vision Technology for Food Quality Evaluation (2nd ed.). San Diego: Academic Press.

About The Authors

James Michael Cello is currently pursuing a Bachelor of Science in Computer Engineering at Eulogio “Amang” Rodriguez Institute of Science and Technology. With a strong interest in Embedded Systems and Software Programming, he contributed primarily to the coding area of this study

Marc Wilson F. Go served as the Builder and Overall Tester for this research. A Computer Engineering student at Eulogio “Amang” Rodriguez Institute of Science and Technology, he was responsible for building and integrating the components into one mechanism. He is passionate about automation and robotics and ensured the technical feasibility of the system.

Lovely B. Nacional served as the Secondary Programmer and Coder, providing critical support in the software development phase. A BS Computer Engineering student at Eulogio "Amang" Rodriguez Institute of Science and Technology, she utilized her interest in smart systems and logic design to assist in implementing algorithms and optimizing the logic flow of the project.

Janette Grace N. Siervo served as the Lead Programmer and Quality Checker for the study. Currently pursuing her BS in Computer Engineering at Eulogio "Amang" Rodriguez Institute of Science and Technology, she leveraged her passion for embedded systems and automation to architect the system's core functionality. She was responsible for maintaining code quality and ensuring the reliability of the system through rigorous testing.

Gian Carlo G. Onia acted as the research leader that oversaw the overall direction, management, and execution of the project. A Bachelor of Science in Computer Engineering student at Eulogio "Amang" Rodriguez Institute of Science and Technology, his leadership ensured the cohesive integration of hardware and software components, guiding the team toward meeting the project’s objectives.

Engr. Bernard C. Fabro is a Professional Computer Engineer and A Professor at Eulogio “Amang” Rodriguez Institute of Science and Technology. He has over 15 years of teaching experience in computer engineering, specializing in robotics, programming, and control systems. His research interests include automation, deep learning applications, and smart systems, with several published works in international conferences and journals.