Department of Computer Science & Engineering
The University of Texas at Arlington

Architectural Design Specification
CSE 4317: Senior Design II
Fall 2025

The Suctioneers
The Vacu-Arm

Andrew Weiler
Andrew Whitmill
Samriddha Kharel
Seezayn Bishwokarma
Jonathan Camarce

Revision History

Revision	Date	Author(s)	Description
0.1	8.22.2025	CC	document creation
0.2	9.19.2025	AW, AW, SK, JC	complete draft
1.0	11.26.2025	AW	edited final draft

Contents

List of Figures

List of Tables

1 Introduction

The Vacu-arm is a robotic arm system designed to automate pick-and-place operations using a suction-based end effector. The system employs the Universal Robots UR20 robotic arm as its primary actuator, integrated with a vacuum suction mechanism to securely grip and manipulate objects up to 50 pounds. The system is controlled through a ROS-based software architecture, enabling precise motion planning, real-time feedback, and emergency safety controls.

The Vacu-arm system consists of three main layers: the Hardware Layer, which includes the UR20 arm and vacuum suction assembly; the Control Layer, responsible for motion planning, ROS node orchestration, and suction control; and the Application Layer, which provides a user command-line interface (CLI) for operation, error logging, and task management.

Key Requirements:

- Support objects weighing up to 50 lbs.
- Execute pick-and-place tasks reliably using vacuum suction.
- Provide emergency stop functionality for safety.
- Allow control through ROS software stack.
- Deliver real-time position feedback and error reporting.

2 System Overview

The overall structure of the software system is designed to seamlessly support the palletizing tasks of the UR20 robot arm system. This layered design enables modular functionality, enhancing system adapability and ensuring each layer performs a unique function while simultaneously providing crucial information to the main PLC layer which manages data distribution. As the central interface, the PLC efficiently processes the flow of sata from each subsystem layer, allowing for smooth system integration and effective coordination across all layers. This architecture promotes reliable, dynamic operation, optimizing the UR20's performance in real-time palletizing tasks.

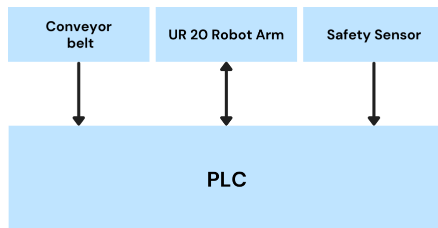

Figure 1: Vacu-arm architectural layer diagram

2.1 Layer 1: PLC Description

The PLC will process most of the data through its peripherals, as seen in Figure 1. It is the primary interface between the different layers of the project. The built-in PLC will be configured with URScript, a Python-based scripting language in which vital functions will reside, such as the input function, the box offset algorithm, and finally, a position algorithm. The data flow will start with the input given by the Photo Eye scanner or the safety sensor, and the input functions will follow these two data paths. First being, an input given by the photo eye scanner will pass through the input function, which will then call the palletizing algorithm and placing boxes in a specific order on a pallet. The safety algorithm will give the second path, which will trigger the input function and force stop the UR20 to create a safe work area for a cooperative application.

2.2 Layer 2: Safety Sensor

The Safety Sensor will determine if a human is in the area of the UR20 arm. If so, it will completely halt the UR20 in order to create a safe work environment. This will be achieved with the use of a camera that will process the data in real-time with the use of computer vision, which will send a signal to the PLC which in turn will send a signal to the UR20 movement subsystem in order to keep the work area safe.

2.3 Layer 3: UR20 Robot Arm

UR20 Arm consists of a vacuum gripper, the gripper controller, and the movement of the arm. The arm is the physical output of the software that resides in the PLC layer. Additionally, it contains a gripper grab/release controller (the controller for the air compressor), which will be turned on or off when needed to hold or drop a box. The commands given by the PLC will determine the position and orientation needed for the UR20 to place the box correctly.

2.4 Layer 4: Conveyor Belt

The conveyor belt system moves over pulleys driven by a motor. As the motor rotates, it propels the belt forward, allowing items placed on the belt to be conveyed along its length. The belt's speed can be adjusted to control the pace of movement. However, for this implementation, it will have two states:

on or off. At the end of the conveyor belt, there will be a guide to reorient the boxes and a bracket in which the Photo Eye scanner will be placed.

3 Subsystem Definitions & Data Flow

Conveyor Belt Layer

- ON/OFF Controller

Safety Sensor Layer

- Computer Vision
- UR20 Arm Speed Controller

UR20 Robot Arm Layer

- Vacuum Gripper
- Grab Release Control
- Movement

PLC Layer

- Input Function
- Box Offset Algorithm
- Position Algorithm

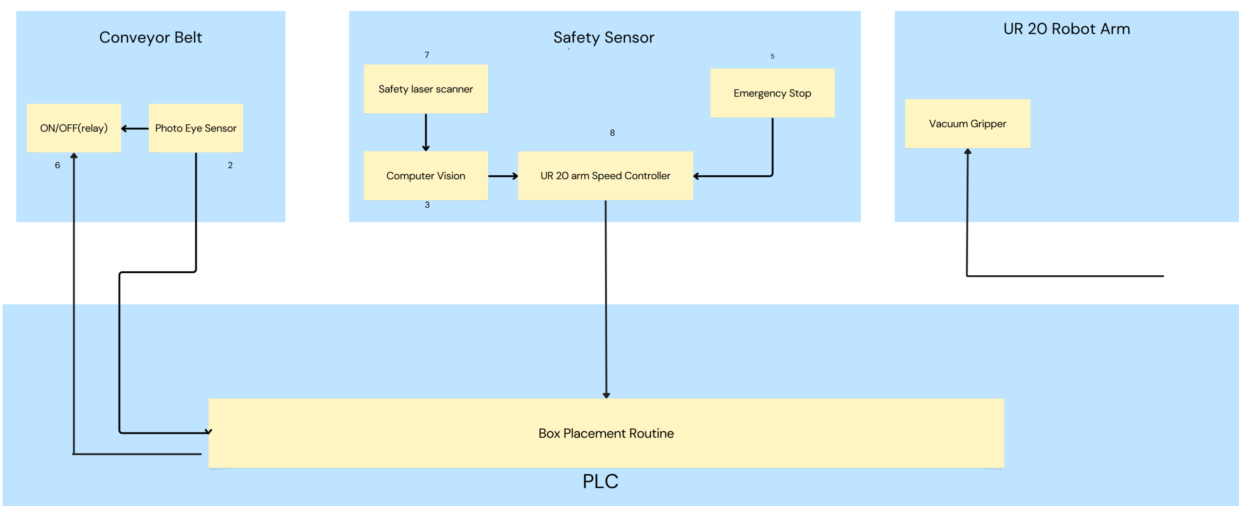

Figure 2: UR20 Data flow diagram

4 PLC Layer Subsystems

The UR20 has a system to practically create a single interface between the robot and PLC, in order to enhance the capabilities and functions of the robot arm.

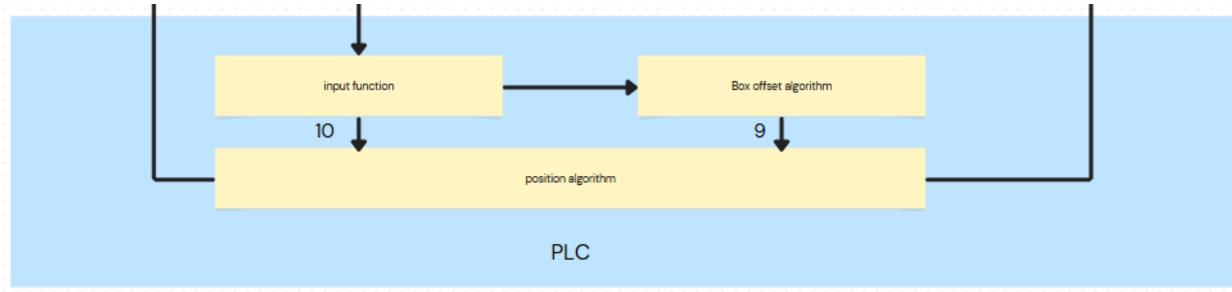

Figure 3: PLC

4.1 Input function

The input function would be the readings of the environment knowing the positions of the pallet and whehere there is a box to pickup based on the photo eye.

4.1.1 Assumptions

Assumptions for the input function is that the orientation of the box is uniform such that it will be picked up the same way.

4.1.2 Responsibilities

The responsibilities of the input function is solely to gather data that the PLC placement algorithm will use to preform its next action. Such as the current state of the pallet, the location of any other nearby objects or people for safety concerns.

4.1.3 Subsystem Interfaces

Table 2: Input fuction interfaces

ID	Description	Inputs	Outputs
#01	movement readings	destination position speeds safety sensor gripper power	formatted data for PLC algorithm
#02	box detection	Photo Eye State position speeds safety sensor gripper power	formatted data for palletizing algorithm

4.2 Box offset algorithm

The box offset algorithm will be used to scan and determine the placement of the box.

4.2.1 Assumptions

The boxes have uniform size and will fit on the pallet with a pre-defined pattern.

4.2.2 Responsibilities

The responsibilities of the placement function will be used to determine location of the box in the x,y,z dimensional plane and send it to the placement algorithm.

4.2.3 Subsystem interfaces

Table 3: Box offset algorithm interfaces

ID	Description	Inputs	Outputs
#03	movement	Photo Eye State position speeds safety sensor gripper power	formatted data for PLC algorithm

4.3 Placement algorithm

The placement algorithm is the central control of operations will tell the subsystems of the UR20 what actions to take given the output of the input function.

4.3.1 Assumptions

The placement algorithm assumes that the gripper will have a fixed strength while moving a box and will be able to place without dropping it. It expects belt and pallet location to be fixed and box orientation constant.

4.3.2 Responsibilities

The responsibilities of the placement function will be to send the appropriate values to the motors, in order to move arm to pick up the box then place it on pallet, and vacuum gripper to pick up and release the boxes. The algorithm will also be checking for safety based on the sensor to scan for people and stop movement.

4.3.3 Subsystem Interfaces

Table 4: Placement algorithm interfaces

ID	Description	Inputs	Outputs
#04	movement	Photo Eye State destination position speeds safety reading	adjusted motor spreeds
#05	gripper power	destination postion	ON/OFF

5 Safety Sensor Layer Subsystems

The UR20 has a Datalogic sensor attached to the PLC that sends in signals if a person or object has entered a zone. The zones can be monitored and changed on the desktop next to the PLC using the DLsentinel application.

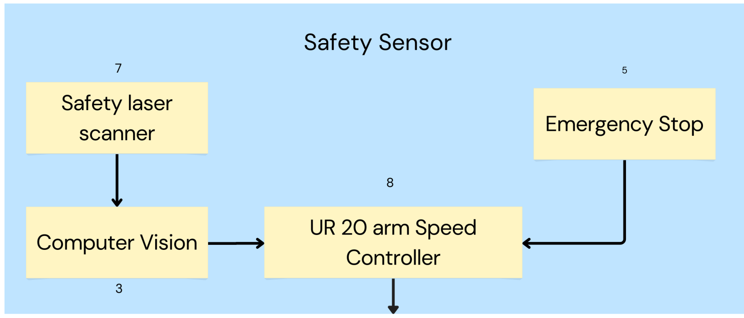

Figure 4: Safety sensor diagram

5.1 UR20 arm halt

The controller will determine whether there is a person within the working vicinity of the UR20 robot and signal the PLC to halt for protection.

5.1.1 Assumptions

The UR20 should have collision detection and should stop at impact.

5.1.2 Responsibilities

To signal the PLC the presence of a person near the robot using a computer vision algorithm.

5.1.3 Subsystem Interfaces

Table 5: Speed Controller Subsystem Interface

ID	Description	Inputs	Outputs
#06	send signal	presence detection	presence signal

5.2 Computer vision algorithm

5.2.1 Assumptions

A computer vision algorithm to scan for the presence of the shape of a person being near the UR20 robot.

5.2.2 Responsibilities

The camera will have a vantage point to be able to see all around the robot.

5.2.3 Subsystem Interfaces

Table 6: Computer Vision Subsystem Interface

ID	Description	Inputs	Outputs
#07	presence detection	camera reading	presence detection

6 UR20 Arm Layer Subsystems

The UR20 arm is a layer controlled by the PLC used exclusively for manipulation. It is a collaborative robot arm that will be used to grab the boxes and position them upon the pallet. It has three subsystems: the movement interface, the vacuum gripper, and the vacuum gripper controller.

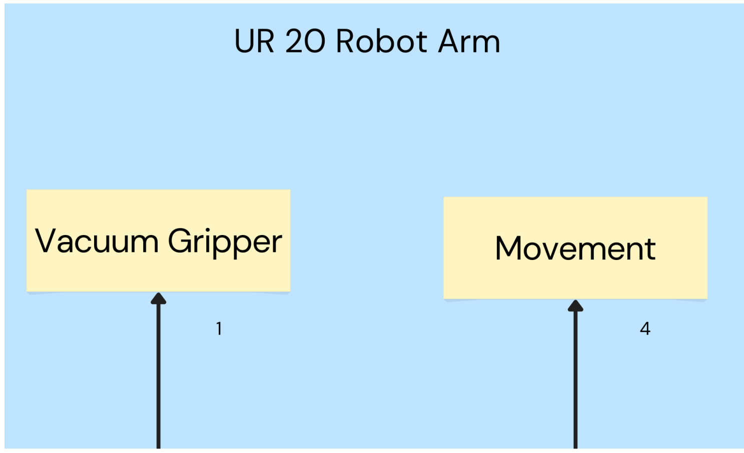

Figure 5: UR20 Robot Arm diagram

6.1 Movement

This subsystem of the UR20 arm is technically an interface within the PLC, but as the PLC and UR20 are connected so closely together, it is listed as a subsystem of the arm. The interface will allow easy control of the arm's movements. Its input is data from the PLC's decision making algorithm and its output is to directly control the robot's movements.

6.1.1 Assumptions

The assumption being made is that the UR20 arm is going to exclusively be a tool of the PLC, and thus will only interface with the PLC. The movement interface may even be built into the PLC for this robot, as it seems to be with some other UR cobots, but as of right now it is being assumed that a control system will have to be programmed for it on the PLC.

6.1.2 Responsibilities

The responsibility of the movement subsystem is to translate the box location, current arm location, and current desired speed into a coherent set of movements for the UR20 arm. These factors will be combined to create a set of movements and movement speed that will move the arm to the desired location.

6.1.3 Subsystem Interfaces

Table 7: Movement interface

ID	Description	Inputs	Outputs
#08	Change in X,Y,Z Coordinates	X,Y,Z	NO OUTPUT
#09	Movement Speed	Speed	NO OUTPUT
#10	Rotation Angles	Perpendicular Longitudinal	NO OUTPUT

6.2 Vacuum Gripper

The vacuum gripper is physical hardware and its software inputs and outputs are covered in the Vacuum Gripper Controller subsystem. The vacuum gripper is connected to a vacuum generator that will be adjusted to provide a static amount of force capable of picking the boxes and not dropping them until the controller determines that it should do so.

6.2.1 Assumptions

The vacuum gripper will be a hardware-only system with a constant vacuum that can be blocked in order to drop a box. It will interface exclusively with the vacuum gripper controller and have no outputs.

6.2.2 Responsibilities

The responsibility of the vacuum gripper is to securely hold and transport boxes. It will be able to drop the boxes on command and in the process it will shift the boxes as little as possible in order to produce the expected placement.

6.2.3 Subsystem Interfaces

The following table summarizes the inputs and outputs for the Vacuum Suction Assembly subsystem.

Table 8: Movement interface

ID	Description	Inputs	Outputs
#11	Vacuum Control Signal	ONOFF	NO OUTPUT

6.3 Vacuum Gripper Controller

The vacuum gripper controller is the software interface between the PLC and the vacuum gripper. It signals the vacuum hardware to kill the vacuum and drop the box being held.

6.3.1 Assumptions

The vacuum gripper controller will be on at all times EXCEPT when it is told that a box needs to be dropped.

6.3.2 Responsibilities

The vacuum gripper controller must turn off the vacuum long enough for the cobot to move away, which should occur soon after. If it turns on again too soon, the box will be pulled out of the desired position.

6.3.3 Movement Subsystem Interfaces

Each of the inputs and outputs for the subsystem are defined here.

Table 9: Movement interface

ID	Description	Inputs	Outputs
#12	Drop Signal	Trigger	ONOFF
#13	Current Speed	Speed	NO OUTPUT

7 Conveyor Belt Layer Subsystems

The conveyor belt layer is managed by the PLC and is solely responsible for controlling the conveyor belts movement. This layer has one subsystem: The ON/OFF controller, which receives signals from the PLC to start or stop the conveyor belt based on when the UR20 robot arm is ready to receive another box. The control of the conveyor belt is simplified to an ON/OFF mechanism, rather than dynamically adjusting speed based on palletizing rate.

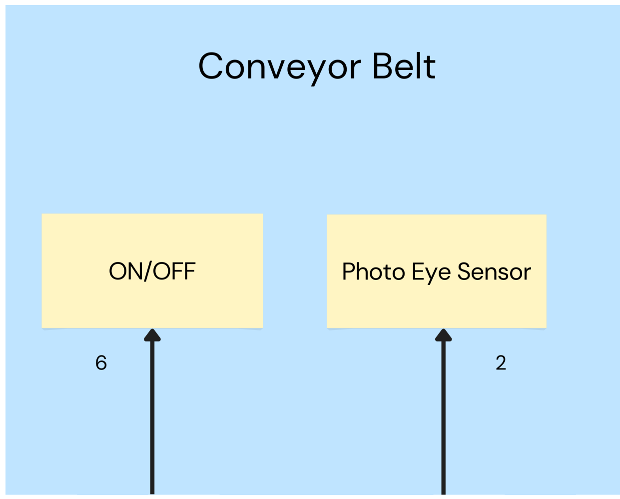

Figure 6: Conveyor Belt Power Controller

7.1 Power Controller Subsystem

The power controller subsystem in the conveyor belt layer is responsible for controlling the state of the conveyor belt. The subsystem receives a command from the PLC to switch the belt "ON" when the robot is prepared to receive a new box and "OFF" once the box has been processed, ensuring smooth and controlled movement of boxes.

7.1.1 Assumptions

The assumption of a required automatic speed control is not required. The belts state change is assumed to be sufficient for the palletizing application, without the need to detect the exact positions of boxes on the belt. Only ensuring that the boxed move in a controlled manner when ready.

7.1.2 Responsibilities

The power controllers main responsibility is to maintain the stability of the conveyor, preventing any from advancing too far or falling off during the palletizing process. By managing the conveyor's ON/OFF state, it ensures that boxes are presented to the robot in a timely and safe manner.

7.1.3 Subsystem Interfaces

The ON/OFF signal from the PLC serves as the sole input for the power controller. When the PLC sends an "ON" signal, the conveyor belt starts moving and will stop movement when the "OFF" signal is received.

Table 10: Controller Interface

ID	Description	Inputs	Outputs
#16	Power Controller signal	ON/OFF	NO OUTPUT

References