Simulation of an autonomous and adaptive robotic arm

Juan Diego Lozada

Cristian Santiago López Cadena

Workshop 2

Universidad Distrital Francisco Jose de Caldas

System Sciences Foundation

Carlos Andrés Sierra

09 de Abril del 2025

Systems Design Framework

System Dynamics Analysis

The system's operation is structured around four fundamental operational phases, which the robotic arm moves through sequentially and with feedback:

1. Environment and object recognition: The agent begins by visually processing the work environment. Using computer vision techniques, the parts present are identified, their relative locations are determined, and morphological classification (size, shape, material, etc.) is performed.

2. Grasp planning and execution: Once the part has been detected and classified, the system estimates the optimal grasping point based on its geometry and center of mass. An initial approach trajectory is generated, which is dynamically updated through sensory feedback (vision and position) to compensate for object displacement. Upon reaching the action zone, the grip is performed, adjusting the force and closing angle according to the pressure sensors.

3. Part transport: The arm calculates the optimal route to the placement destination. During this journey, the force exerted on the object is monitored in real time, ensuring it remains within safe thresholds to prevent damage or falls, and adjustments are made as variations are detected.

4. Release and verification: Upon reaching the delivery point, the system executes the object release sequence. A return trajectory to the resting point is then planned, but first, it positions itself to capture an image of the deposited object. This image is used as feedback to verify whether the part was placed correctly and in optimal conditions, thus informing the agent's learning process for future tasks.

Phase diagrams

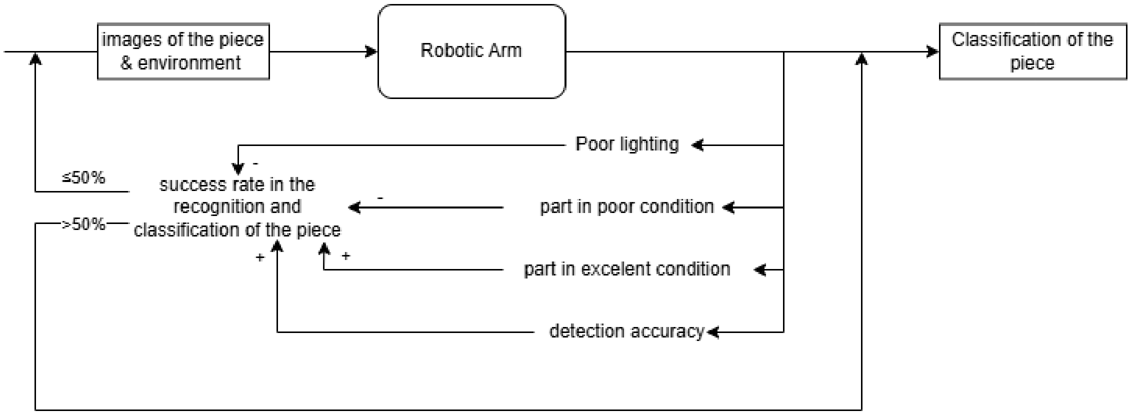
images of the piece & environment
Robotic Arm
Classification of the piece
Poor lighting
-
≤50%
success rate in the recognition and classification of the piece
-
part in poor condition
>50%
+
+
part in excelent condition
detection accuracy

Figura 1

Environment and object recognition phase

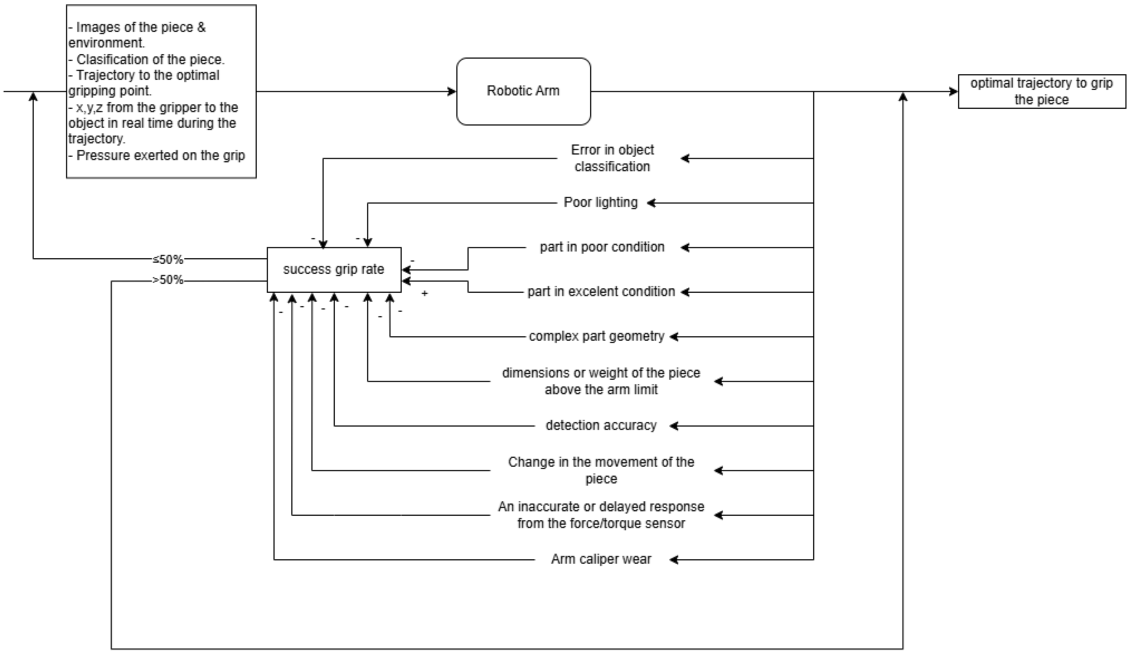
- Images of the piece & environment.
- Clasification of the piece.
- Trajectory to the optimal gripping point.
- x,y,z from the gripper to the object in real time during the trajectory.
- Pressure exerted on the grip
Robotic Arm
optimal trajectory to grip the piece
Error in object classification
Poor lighting
part in poor condition
part in excelent condition
complex part geometry
dimensions or weight of the piece above the arm limit
detection accuracy
Change in the movement of the piece
An inaccurate or delayed response from the force/torque sensor
Arm caliper wear
success grip rate
≤50%
>50%
-
-
-
+
-
-
-
-
-
-

Figura 2

Grasp planning and execution phase

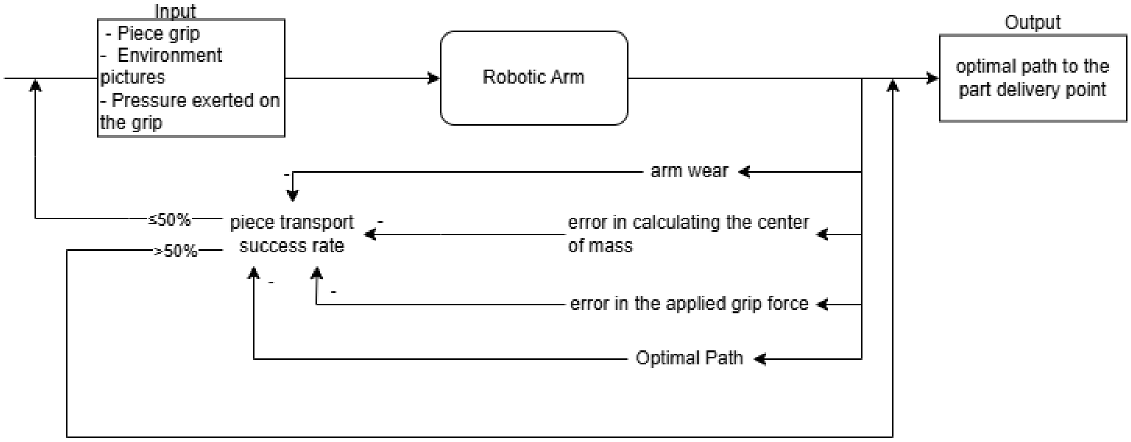
Input
- Piece grip
- Environment pictures
- Pressure exerted on the grip
Robotic Arm
Output
optimal path to the part delivery point
arm wear
-
≤50%
>50%
piece transport success rate
-
error in calculating the center of mass
-
-
error in the applied grip force
Optimal Path

Figura 3

Part transport phase

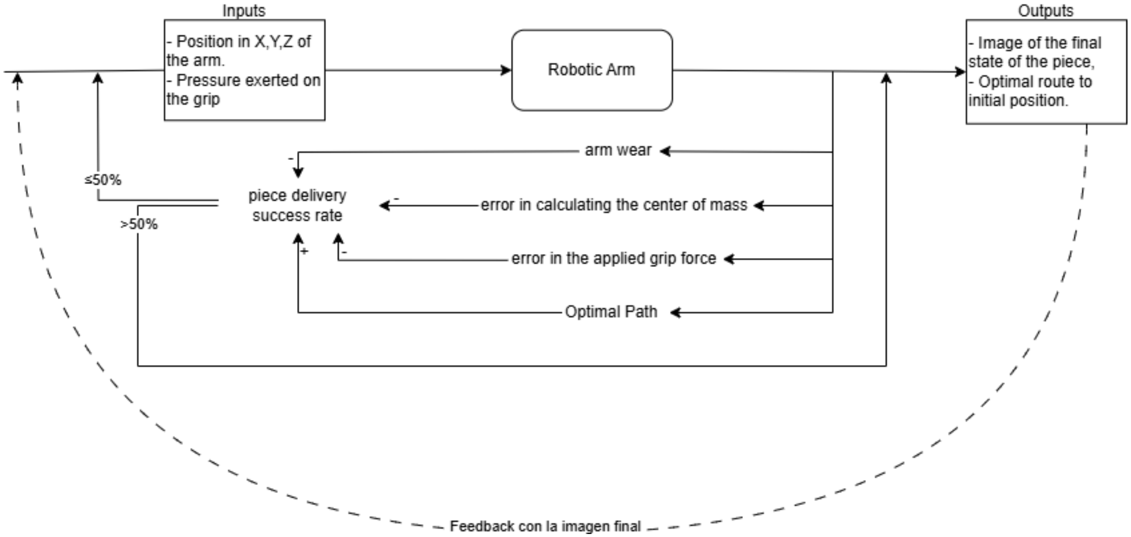
Inputs
- Position in X,Y,Z of the arm.
- Pressure exerted on the grip
Robotic Arm
Outputs
- Image of the final state of the piece,
- Optimal route to initial position.
arm wear
-
≤50%
>50%
piece delivery success rate
-
error in calculating the center of mass
+
-
error in the applied grip force
Optimal Path
Feedback con la imagen final

Figura 4

Release and verification phase

As evidenced in the previously presented phase diagrams, multiple factors have been identified that can negatively or positively affect the success rates associated with each phase of the system's operation. The following performance metrics were also defined:

1. Piece recognition and classification success rate (Phase 1, Figura 1).

2. Piece grip success rate (Phase 2, Figura 2).

3. Piece transport success rate (Phase 3, Figure 3).

4. Piece delivery success rate (Phase 4, Figura 4).

For each phase, performance thresholds were established that allow for adaptive decisions. If the success rate is less than or equal to 50 %, the system activates feedback mechanisms to reanalyze inputs and collect new data. If it exceeds this threshold, the system continues executing the current phase's objective. These initial thresholds are considered provisional and will be progressively optimized using reinforce learning techniques, aiming to achieve a success rate equal to or greater than 80 %.

Mathematical/Simulation Model. For this project, we chose to implement the Soft Actor-Critic (SAC) algorithm due to its approach based on nonlinear optimization models and stochastic policies, making it especially suitable for dynamic and uncertain environments such as those encountered in robotic arm control. This algorithm is based on the formulation of the smoothed Bellman equation, integrating an entropy term that maximizes both the expected reward and exploration, which is essential for the agent's continuous adaptation to variations.

$$J(\pi_{\theta)} = E_{\pi_\theta}[\sum_{t=0}^{T-1} \gamma^t R(s_t, a_t) + \alpha H(\pi(.|s_t))] \tag{1}$$

Donde:

- $J(\pi_{\theta)}$ It is the policy that is being learned
- E_{π_θ} Mathematical expectation on the trajectories generated by the policy π_θ
- γ^t Discount factor that reduces the importance of future rewards
- $R(s_t, a_t)$ Reward received upon performing the action a_t in the state s_t.
- $H(\pi(.|s_t))$ represents the entropy of the policy π in the state s_t.

- α Coefficient that controls the importance of entropy in relation to reward.

This approach is particularly useful for addressing uncertainty factors and success rates identified in the system's phase diagrams, allowing the agent to make decisions that balance the exploitation of successful policies with feedback from new observations, achieving progressive improvement in the execution of complex tasks such as object identification, grasping, transport, and release. The implementation of this algorithm will be carried out for each phase as follows:

1. Environment and Part Recognition Phase: In this phase, the SAC algorithm allows the system to learn to efficiently select the most relevant areas even in changing environments.

   - Input: Images of the environment and the part.
   - Action: Selection of areas of interest for analysis, identification of patterns in the image.
   - Reward: Awarded if the object is correctly identified and classified

2. Object Grasping Phase: In this phase, the implemented policy allows the user to learn how to choose the optimal route and optimal forces based on the type and shape of the object. Entropy is used to promote the exploration of gripping alternatives.

   - Input: Images of the object, object classification, object coordinates, trajectory to the optimal grasping point, force applied to the part.
   - Action: Determination of the grasping point, movement to the grasping point, and determination of the force to be applied for the grasp.
   - Reward: Successful grasping without damaging or losing the object.

3. Object Transport Phase: In this phase, the SAC algorithm is implemented to adapt the route to obstacles or misalignments detected in real time.

- Input: Current point and destination. Pressure and acceleration sensors, trajectory.
- Action: Safe trajectory selection and dynamic adjustments.
- Reward: Minimizing path deviation and maintaining stable gripping conditions.

4. Release and Final Position Verification Phase: In this phase, the policy can improve the release process based on visual and sensor feedback to increase the success rate.

   - Input: Images or coordinates of the target location, grip force.
   - Action: Release of the object and fine-tuning of positioning.
   - Reward: Placement accuracy and matching with the target state.

Feedback Loop Refinement

Enhanced Control Mechanisms.

After reviewing the sensors, we decided to install an RGB-D camera in the handle. This sensor not only allows for an in-depth analysis of the object in front of the robot, but it also allows us to identify the object's trajectory in real time and validate the object's condition once it's delivered. We also decided to incorporate acceleration sensors in the arm structure to adapt the speed to the transport process. On the other hand, we identify specific actions for each phase to assign more granular rewards so the arm can obtain more immediate feedback:

1. Environment and Part Recognition Phase: If the object is detected (+0.5). If the object is not detected and its presence has been confirmed (-0.3). If the object is classified correctly (+0.5). If the classification is incorrect (-0.5). If the object classification is correct and matches a previous one (+0.2). (-0.1 * time) we can subtract 0.1 for the time it takes to identify the object, forcing it to be more efficient. If it manages to follow the trajectory of an object (+0.5). If it loses the identified object (-0.5). If it identifies objects disproportionate to its capabilities (+1).

2. Object Grasping Phase: If the trajectory to the grasping point is optimal (+0.3). If the force applied was correct (+0.5). If the piece falls or is deformed due to excessive force (-1). If the grip is outside the optimal point (-0.5)

3. Object Transport Phase: Penalty for long trajectories (- path length). If the speed changes suddenly (-0.3). If the part falls (-1). If the part ends at the delivery point (+0.5).

4. Release and Final Position Verification Phase: If the part is in the correct position and orientation for release (+1). If it is far from the release point (-d), which is the distance from the correct position. If the arm is not repositioned correctly to capture an image of the delivery state (-0.6). If it is positioned correctly (+0.5). If it damages other parts during release (-0.1).

Stability and Convergence.

Some criteria for agent stability are:

- Its behavior must be well-constrained; the arm should not attempt to grasp objects outside its range of action, apply excessive force to the grip, or move at excessive speed. These parameters, when constrained, incorporate the BIBO stability of cybernetic control.

- The success rates defined for each phase should show sustained improvement. Furthermore, the policy and value function must stabilize over time as the arm performs similar tasks (grasping, moving, releasing) in a repeatable and consistent manner.

Iterative Design Outline

Project Plan.

- The main purpose is improve the project in a simple and efficient way, so, to support our autonomous agent we will use the next things, We will use a projection matrix to convert 3D world coordinates into 2D image coordinates based on the intrinsic (describe the intern part from the camera and how it show the points of three-dimensional space to the image plane) and extrinsic (describe the position and the orientation from the camera in a 3D) parameters of the RGB-D camera. This enables accurate localization of objects in the camera frame, crucial for computing grasp points and motion planning.

The intrinsic Parameters are the next one:

- Focal Length (f)
- Principal Point (cx,cy)
- Skew, inclination factor

$$K = \begin{pmatrix} f_x & s & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{pmatrix} \tag{2}$$

Extrinsic Parameters

The transformation from a 3D point in the world (X, Y, Z) to a 2D point in the image (u, v) is:

$$s \begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = K \begin{bmatrix} R & t \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix}$$

RGB-D frames will be collected during simulation and real-world trials, consisting of synchronized RGB and depth images. These will be annotated with class labels, center-of-mass coordinates, and segmented masks. Calibration data (camera intrinsics, homographies, and transformation matrices) will also be stored to support 3D reconstruction and labeling.

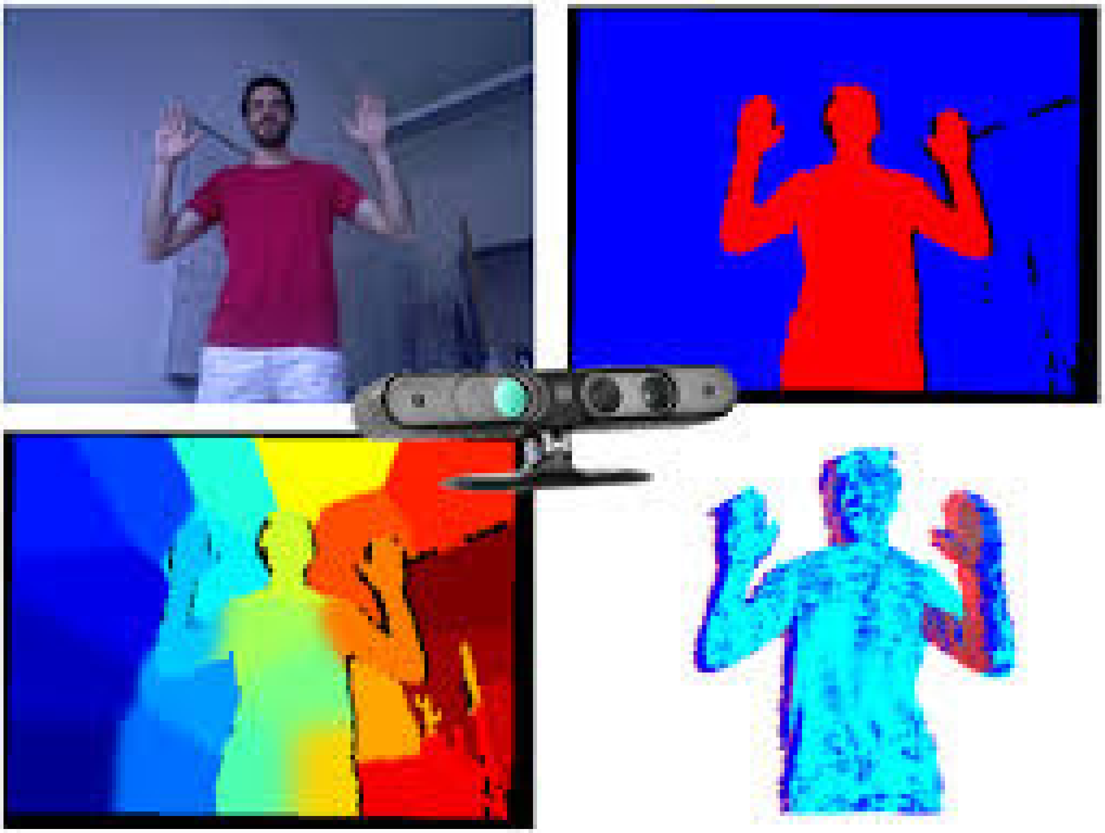

Figura 5

Camera RGBD

To train the robotic agent, we plain to use Soft Actor Critic (SAC), a cutting-edge deep reinforcement learning algorithm that's known for being both stable and efficient with data. Since it's an off-policy method (that means the program could learn using data from the past), which is super helpful when working with real robots where collecting data can be slow or expensive, making cheaper the training.SAC is especially good for continuous (could take infinites inside a range) control tasks, like adjusting grip angles or the amount of force the robot applies when picking something. One of the key features that sets it apart is entropy regularization, it encourages the robot to keep exploring different actions instead of making the same exit. This helps the robot discover better, more flexible and behaviors over time.

Testing and Simulations

As we mentioned on the previous update *Workshop1*, we still have to idea to implement the same tools for the testing, due to, those are the most safe and stable, such as Open GymIA, PyBullet, Stable-Baselines3. Knowing the diferent outputs or random events could happen, those frameworks are useful, for instance, if we want to make a scene with success rate as we show at the beginning of the document

We're going to train the agent several times using different random seeds. This is important because reinforcement learning training depend on how it starts. By using multiple seeds, we'll be able to see if the agent's behavior is consistent or if it just performs well due to luck. This will help us understand how robust the model is in slightly different situations.

In each training or testing cycle, we will record several performance indicators, such as how often it successfully grasps the object (grasp success rate), efficient the arm's trajectory is (trajectory smoothness), and how long it takes to make the classification. If it frequently fails to grasp objects, we could adjust the reward function or change the sensors it's using. These indicators will be key in a feedback loop that will allow us to continuously improve the agent. And taking account we cannot guarantee a hundred percent but we can maintain a range

Referencies

Haarnoja. (2018). Soft actor-critic: Off-policy maximum entropy deep. *Cornell University*. Descargado de https://arxiv.org/abs/1801.01290v2

Santos, A. (2016). Projection matrix. *Science Direct*. Descargado de https://www.sciencedirect.com/science/article/abs/pii/S0888327015004938

Haarnoja (2018) Santos (2016)