



BERKSHIRE
GREY®



WHITE PAPER

The Right Automation Boundary

How Much Should a Picking Robot Actually Do?

By Sanath Vurelli, Senior Product Development Engineer at Berkshire Grey

| A system-level framework for robotic piece-picking architecture

Robotic picking projects often begin with a straightforward objective: automate the movement of an item from a source container to its destination. But the scope rarely stays that simple.

Should the robot also exchange totes, scan barcodes, pack cartons, print labels, or resolve exceptions? Each additional task may be technically possible. Each also consumes cycle time, adds complexity, and changes the economics of the complete system.

The line between the work assigned to the robot and the work handled by surrounding automation, software, or people is the automation boundary. Where that boundary sits can affect throughput, labor, capital cost, resilience, and ultimately return on investment.

The objective is not maximum autonomy. It's to use robotic capacity where flexibility, perception, and dexterity create the most value, while assigning other tasks to the resources that can perform them most economically.

Four questions frame the decision:

- Does the task require the robot's flexibility, perception, or dexterity?
- How much robot capacity does it consume under realistic conditions?
- What labor, equipment, floor space, or downstream process does it eliminate?
- Could another technology perform it with lower total cost or operational risk?

The sections that follow show how to apply these questions at the system level, where completed orders, operating resilience, and total economics matter more than the number of features assigned to the robot.

Robot Cycle Time is Scarce Capacity

Industrial robots are attractive because they combine repeatable motion with flexible manipulation. With the right perception, gripping, and planning, one robot can handle a changing mix of products that would be difficult to process with fixed mechanisms. That flexibility is valuable, but it's also expensive. It should be used where the operation actually needs perception, dexterity, or product-level decision making.

Many surrounding tasks are more deterministic. A conveyor can move a carton through a known path. An automated storage and retrieval system (ASRS) can present an inventory bin. A shuttle can exchange destination containers. Software can sequence work, manage buffers, and decide which order should be

released next. A person can resolve an unusual packaging exception without forcing the automated cell to support that exception on every cycle.

If the robot performs those tasks, they consume the same motion resource needed for picking. That creates an opportunity cost.

The simplest way to see the tradeoff is to treat a robot's available operating time as a capacity budget. In one hour, it has 3,600 seconds to divide among every task assigned to the cell.

Use flexible robotic capacity where flexibility creates value. Use simpler automation for repeatable material movement when it can do the job more economically.

A Six-Line Order Shows the Tradeoff

Consider a six-line order in a goods-to-person ASRS operation. Assume the cell requires an average of 10 seconds per line across presentation, picking, and placement, with no overlapping activity. The robot therefore spends about 60 seconds completing the six picks.

Now assume the cell needs another 30 seconds to finish the order, remove the completed destination tote, bring in an empty tote, verify its position, and prepare for the next order. Total robot-side processing time becomes 90 seconds per order.

At 90 seconds per order, the theoretical capacity is 40 orders per hour. If destination-tote exchange is moved to another resource and the robot can return to picking after 60 seconds, theoretical capacity rises to 60 orders per hour.

HOW TASKS CAN IMPACT CAPACITY

Robot task	Time per order	Theoretical capacity
Picking & tote exchange	90 seconds	40 orders per hour
Picking only	60 seconds	60 orders per hour

These numbers are illustrative only, and may not reflect actual performance.

That's a 50 percent increase in robot-side order capacity. The robot has not become faster at grasping, perception has not improved, and the arm has not been replaced. The system simply stopped using one-third of the robot's time for non-picking work.

This doesn't automatically mean tote exchange should be assigned elsewhere. The additional motion may eliminate labor, a shuttle, or another piece of equipment. It means that the capacity cost, and what the operation receives in return, must be visible.

Real operations include presentation delays, failed grasps, changing SKU difficulty, order variability, and recovery time. Still, the underlying tradeoff remains: a secondary process can consume a large share of capacity even when each motion appears small.

This also explains why peak pick rate can mislead. A robot may demonstrate a fast pick-to-place cycle and still produce fewer completed orders because it spends substantial time managing containers or waiting. A cell with a modest peak rate can perform well if it receives work consistently and avoids unnecessary motions.

Change the automation boundary and the robot's job can become simpler: pick, place, repeat. The robot performs fewer distinct processes, but the operation may produce more completed work.

| Autonomy and productivity are not the same thing.

Start with the System Constraint

Saving robot cycle time only creates value when that time affects the performance of the system. This is where an otherwise sound optimization can become irrelevant.

Imagine an ASRS that can present only 200 pick opportunities per hour to a cell. If the robot can already process 230, increasing its capacity to 260 will not raise system throughput. The robot will simply wait longer for inventory. In that case, removing a few seconds of robot motion may still improve robustness or create room for future growth, but it will not increase current output.

Now consider the opposite case. The ASRS can support 300 pick opportunities per hour, but the robot can process only 230 because container exchange and packing motions consume too much time. The robot is the constraint. Recovering those seconds can translate directly into more throughput, at least until another resource becomes limiting.

For that reason, the right question is not merely, "How many seconds does this feature add?" It's, "What happens at the system constraint when this feature is added?"

If a robot task consumes capacity without removing another constraint, the system may slow down without an offsetting benefit. If it eliminates an expensive downstream process, the same task may improve total economics even as robot picks per hour decline.

Throughput should be evaluated at the level customers care about: completed orders, cartons, or shipped units. Robot picks per hour is an important engineering metric, but it's an input to the business result, not the result itself.

Three ways to measure capacity:

- 1. Theoretical capacity:** What the cycle-time calculation permits.
- 2. Effective capacity:** What the system produces after availability losses, failed attempts, blocked time, and normal variation.
- 3. Business capacity:** The completed orders, cartons, or shipped units the operation actually realizes.

A feature may reduce theoretical robot rate but increase effective output by preventing stops. Another may raise peak rate without changing business capacity because the robot is usually starved. Separating these measures makes those outcomes visible.

Decouple Picking From Packing

One of the most useful design choices is to separate processes that don't need to run at the same instantaneous rate. Picking and packing are a common example.

If every completed pick must immediately flow through cartonization, packing, labeling, and shipping preparation, then a delay in any downstream step can interrupt the robot. The entire chain becomes tightly coupled. A printer issue, a carton shortage, a difficult pack, or a brief operator delay can propagate upstream and leave robotic capacity unused.

A buffer changes that relationship. The robot can build a controlled amount of completed work while downstream resources consume it at a different rate, absorbing short-term variation.

This doesn't remove the need to balance average capacity. If the robot produces 300 completed orders per hour and pack-out can sustain only 200, a finite buffer will eventually fill. But a buffer can protect the robot from normal variability: a few unusually difficult cartons, a temporary pause, an uneven order mix, or a short replenishment delay.

Decoupling also gives teams more freedom to schedule labor and recover from minor disruptions. Picking can continue while downstream work catches up, provided order-release rules, traceability, and buffer capacity support it.

Buffers require floor space, container inventory, tracking, controls, and exception handling. Excessive work in process can increase congestion. The goal is to use a deliberate decoupling point where synchronization is unnecessarily reducing utilization.

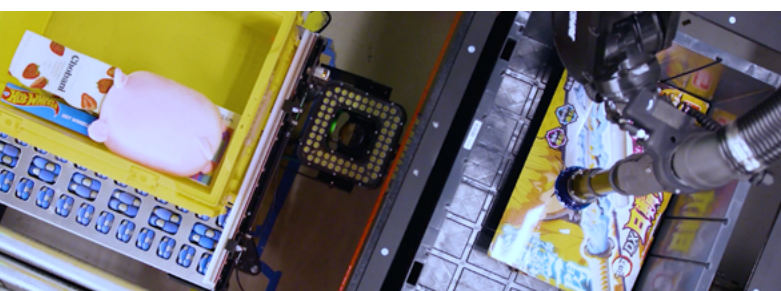
Prevent downstream processes from interrupting robotic picking unless the process truly requires them to remain synchronized.

Put Each Task on the Right Technology

A fulfillment system may combine robotic arms, conveyors, ASRS equipment, autonomous mobile robots (AMRs), shuttles, scanners, software, packing automation, and people. The goal is not to maximize the role of any one technology. It's to assign each task to the resource that performs it most economically while meeting throughput, quality, safety, and flexibility requirements.

The distinction between flexible and deterministic work is a useful starting point.

Picking an unknown item from a cluttered bin can require perception, grasp selection, collision avoidance, and adaptation to product variation. That's a good use of a robotic arm. Moving a tote along the same fixed route each cycle may be better suited to a conveyor or transfer mechanism. Deciding which order should enter the cell next is usually a software problem. Handling a rare, poorly packaged, or damaged item may remain a human exception because automating it would burden every cell with hardware and logic used only occasionally.





The cheapest component isn't always the cheapest system choice. A transfer mechanism may reduce robot cycle time but add a maintenance point. An AMR may improve layout flexibility but create traffic and charging requirements. A person may be economical at low volume but difficult to staff around the clock.

FOUR QUESTIONS TO FRAME AUTONOMY

- **Does the task require the robot's flexibility, perception, or dexterity?**
- **How much robot capacity does it consume under realistic conditions?**
- **What labor, equipment, floor space, or downstream process does it eliminate?**
- **Could another technology perform it with lower total cost or operational risk?**

Sometimes Asking the Robot to Do More Creates Value

A narrow robot scope is not automatically better. There are operations where asking the robot to do more creates stronger economics.

Direct-to-carton picking is a good example. An ASRS presents inventory, and the robot places each item directly into the final outbound carton. This can add significant requirements. The cell may need to consider available carton space, product orientation, fragility, packing sequence, fill height, weight distribution, and the risk of placing a heavy product on a fragile one. Perception and motion planning become more difficult. Placement may take longer than releasing an item into an intermediate tote.

Measured strictly at the robot, the cell may appear slower. Yet direct-to-carton placement can eliminate a downstream transfer and packing step. It may remove labor, reduce touches, lower floor-space requirements, and avoid the equipment needed to move completed picks to another station. The slower robot can produce the better fulfillment system.

The same logic can support robot-managed container exchange. If an operation has low order volume, limited space, and no economical way to add a separate tote-handling mechanism, using spare robot time may be sensible. A highly utilized transfer device would be hard to justify, while the robot may already have enough unused capacity to perform the exchange without constraining output.

The decision comes down to net economics. A task shouldn't be rejected because it makes the robot slower. It should be evaluated against what the task removes and how it changes the rest of the process.

Complexity Carries an Operating Cost

Cycle time is the most visible cost of expanding robot scope, but it's not the only one.

Every added capability creates states the system must recognize and manage. A container may arrive late, arrive in the wrong orientation, fail to scan, or become stuck. A carton may be the wrong size. A label may not print. A product may not fit after earlier placements. A gripper may lose an item while the destination is being exchanged. Each event needs detection, a recovery path, operator guidance, and a safe way to resume.

These interactions grow faster than the feature list suggests. A pick-and-place cell has a relatively clear loop. A cell that also handles containers, packing constraints, labeling, and order closure must coordinate more devices and shared state. The hard part is making delays and exceptions recover predictably in production.

Complexity also affects deployment. More interfaces must be commissioned, more fault cases tested, and more people trained. Root-cause analysis becomes harder because a throughput loss may originate in robotics, material flow, order data, packaging, or their interaction.

An apparently small feature can therefore have a large lifecycle cost. The review should include software, controls, safety, commissioning, maintenance, and operations. A few seconds of extra cycle time may be acceptable while frequent manual recovery is not.

Different Automation Boundaries Solve Different Problems

Robotic picking architectures can sit at several points on an automation spectrum; each solves a different operational problem.

ROBOTIC PICKING ARCHITECTURES

Architecture	What it can optimize	Primary tradeoff & fit
Pick & release	Protects manipulation capacity and keeps the robotic loop relatively simple.	Leaves more downstream handling; often strongest when the robot is the constraint.
Pick & pack	Can reduce downstream touches, labor, and transfers.	Packing rules and controlled placement can consume additional robot time.
Pick, pack & container handling	Can reduce reliance on separate tote-handling equipment.	Adds non-picking motion and recovery states; may fit lower-volume or space-constrained operations with spare robot capacity.

The right design depends on order profile, SKU characteristics, throughput, labor model, floor space, available material-handling equipment, uptime requirements, and where the process is constrained.

The boundary can change over time. An initial deployment may use the robot for container exchange because volume is low. As volume grows, a dedicated transfer system may let the robot focus on picking. Elsewhere, improvements in packing perception may make direct-to-carton placement worthwhile later.

A modular architecture avoids forcing every capability into the first release.

Evaluate the Complete Economics

Robotic picking ROI cannot be reduced to peak picks per hour, and it cannot be reduced to labor replacement alone. Both matter, but neither captures the full system.

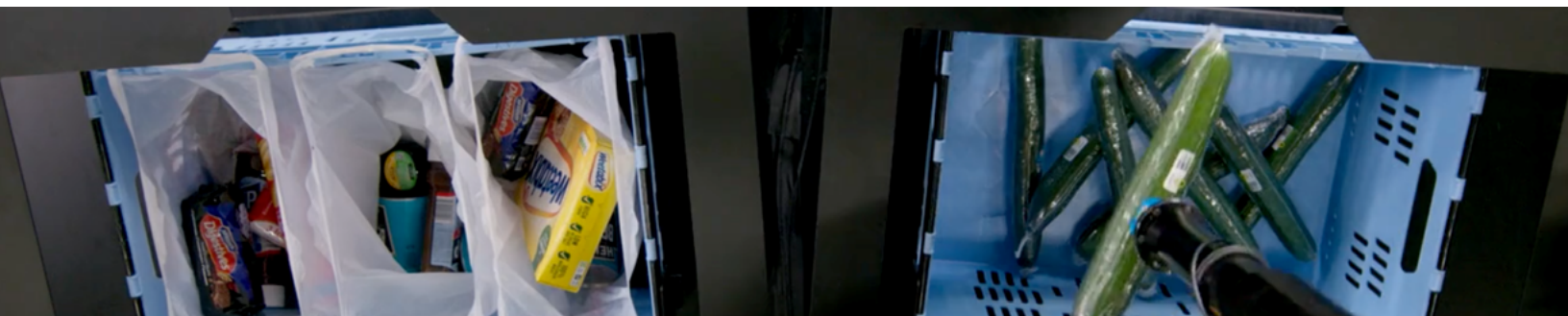
The evaluation should include utilization, supporting automation, availability, buffers, floor space, downstream touches, container inventory, maintenance, interventions, and the process eliminated. It should consider variability, not only averages.

A practical comparison can be built around a small set of measures:

- **End-to-end throughput:** completed orders or cartons per hour, not just robot motions.
- **Constraint utilization:** how the design changes the workload of the current bottleneck.
- **Labor content:** routine staffing plus exception handling, replenishment, and recovery.
- **Capital and lifecycle cost:** robot, peripheral equipment, controls, integration, maintenance, and future changes.
- **Operational resilience:** the impact of faults, blocked downstream processes, order variability, and staffing gaps.
- **Process removal:** touches, queues, stations, equipment, and floor space that disappear from the final design.

Compare these measures across realistic scenarios: high-line-count and single-line orders, difficult SKUs, uneven presentation, downstream interruptions, and order transitions.

A design that looks efficient at a steady average may struggle when the mix changes.



Turn the Analysis into an Architecture Decision

The automation boundary should be decided before the cell design becomes fixed, then revisited as better data becomes available.

- 1. Map the complete process.** Document every task, wait, handoff, verification, and exception from inventory presentation through outbound release. Don't begin by assigning tasks to the robot.
- 2. Measure time and variability.** Separate product manipulation from container movement, scanning, packing, labeling, and recovery. Even rough measurements can reveal that a seemingly small task consumes a meaningful share of the cycle.
- 3. Identify the system constraint.** Determine what limits output today and what's likely to limit it after automation. Moving work away from the robot may simply move the bottleneck downstream, which can still be appropriate if downstream capacity is easier or less expensive to add.
- 4. Calculate what each architecture costs and removes.** Account for robot capacity, labor, equipment, commissioning, maintenance, floor space, and downstream touches.
- 5. Test exceptions, not just the happy path.** Ask what happens when a scan fails, a buffer fills, a carton is unavailable, an item doesn't fit, or a downstream process stops. Favor the architecture that continues to create useful output and recovers without excessive intervention.

The Right Level of Autonomy

So, how much should a picking robot actually do? There is no fixed answer. The scope of the robot's role should reflect where the system is constrained, which tasks require flexible robotic manipulation, and what each added capability removes elsewhere in the operation.

The goal isn't maximum autonomy. It's the right autonomy: enough to remove costly work and operate reliably, but not so much that valuable robot capacity is consumed by tasks better handled by other automation, software, or people.

That's ultimately how a robotic picking architecture should be judged.

The best robotic cell is not the one that performs the most tasks. It's the one that creates the most economic value at the system level.



About Berkshire Grey

Berkshire Grey, a SoftBank Group wholly owned subsidiary, is a global leader in AI-driven robotics that transforms warehouse and fulfillment operations for parcels and logistics providers, 3PLs, retailers and ecommerce customers. The company provides end-to-end automation solutions at scale across critical, high-volume workflows, from picking and sortation through dock automation. Powered by deep expertise in robotics, artificial intelligence, and software, Berkshire Grey enables customers to reduce labor risk, increase throughput, and flex operations in response to rapidly changing demand.