The Convergence of Embodied Intelligence: An Exhaustive Technical Analysis of Physical Intelligence π0.6 and Dyna Robotics DYNA-1

Executive Summary

The field of robotics is currently navigating a paradigm shift of historical magnitude, transitioning from the modular, hand-engineered software stacks that have defined the discipline for fifty years to monolithic, learning-based foundation models. This report provides an exhaustive technical analysis of two leading architectures driving this transition: the Vision-Language-Action (VLA) models developed by Physical Intelligence (specifically the π_0, $\pi_{0.6}$, and $\pi^*_{0.6}$ architectures) and the Reinforcement Learning (RL)-centric foundation models developed by Dyna Robotics (DYNA-1).

The analysis reveals a fundamental divergence in philosophy and technical execution between the two entities. Physical Intelligence pursues a "Generalist" strategy, leveraging internet-scale pre-training, novel Flow Matching policies, and a "System 1" intuition-based architecture to create a universal robotic brain capable of zero-shot generalization across diverse hardware configurations.[1] This approach collapses the traditional perception-planning-control hierarchy into a single neural network pass, achieving perception and planning "in one shot" through high-dimensional latent space manipulation rather than explicit search.

In contrast, Dyna Robotics adopts a "Specialist-to-Generalist" strategy, deeply rooted in the proprietary application of Large Language Models (LLMs) for automated reward engineering (*Eureka*) and Sim-to-Real transfer (*DrEureka*). This approach prioritizes commercial-grade reliability, achieving success rates exceeding 99.4% on long-horizon dexterous tasks by solving the physics of manipulation in simulation before deploying to the real world.[4]

This report details the mechanisms of perception and planning in the $\pi_{0.6}$ architecture, distinguishing its inference capabilities from traditional hierarchical planning. It further

scrutinizes the sensory inputs—notably the exclusion of Lidar in favor of vision-centric navigation—and the specific control frequencies and latency management strategies employed to make these large models distinctively "physical." Finally, it juxtaposes these modern approaches against the traditional robotic stack, highlighting the implications for the future of industrial and domestic automation.

---

1. The Paradigm Shift: From Modular Stacks to Foundation Models

To fully appreciate the architectural innovations of Physical Intelligence and Dyna Robotics, one must first understand the incumbents they seek to displace. The history of robotics has been dominated by the "Modular Stack," a methodology that decomposes the problem of agency into distinct, mathematically tractable sub-problems.

1.1 The Traditional Robotic Stack: The Baseline

The traditional approach, exemplified by the Robotic Operating System (ROS) ecosystem, treats robotics as a pipeline of isolated functions. This pipeline typically flows from Perception to State Estimation, then to Planning, and finally to Control.

1.1.1 Explicit Perception and State Estimation

In a traditional system, perception is the process of mapping sensor data to a geometric representation of the world. A robot equipped with Lidar and cameras uses algorithms like SLAM (Simultaneous Localization and Mapping) to construct a voxel grid or point cloud. This requires "State Estimation," often utilizing Kalman Filters (EKF/UKF) to fuse noisy sensor data into a coherent estimate of the robot's position ($x, y, z$) and orientation (quaternion). This process is explicit and geometric; the robot "knows" where the obstacle is because it has calculated the obstacle's coordinates relative to its base frame.

1.1.2 The Deliberative Planning Layer

Once the state is estimated, the "Planning" layer takes over. This is traditionally a "System 2" process—slow, deliberative, and calculating.

- **Global Planning:** Algorithms like A* or Dijkstra search a graph to find a path from point A to point B.
- **Motion Planning:** For a robotic arm, algorithms like RRT (Rapidly-exploring Random Trees) or PRM (Probabilistic Roadmaps) search the high-dimensional configuration space (C-space) to find a sequence of joint angles that moves the end-effector to the target without self-collision.
- **Trajectory Optimization:** Algorithms like CHOMP or STOMP smooth this path into a feasible trajectory, minimizing jerk and energy.

1.1.3 The Control Loop

Finally, the "Control" layer executes the plan. A PID controller or Model Predictive Control (MPC) loop runs at high frequency (1 kHz) to calculate the exact motor torques required to track the planned trajectory, correcting for real-time disturbances.

1.1.4 The Failure of the Modular Approach

While mathematically elegant, this stack suffers from "compounding errors." If the perception layer misclassifies a shiny object as free space (a common failure with Lidar/Depth cameras on reflective surfaces), the planner will confidently plot a course through the object, and the controller will flawlessly execute a collision. Furthermore, this approach is brittle in "unstructured" environments. A "messy room" is a geometric nightmare for a classical planner; it presents too many variables, too many unknown objects, and too much semantic ambiguity (e.g., a blanket can be driven over, a box cannot). The "Bitter Lesson" of AI suggests that hand-engineering these geometric heuristics is ultimately inferior to learning them from massive data.[6]

---

2. Physical Intelligence: The Vision-Language-Action (VLA) Architecture

Physical Intelligence (PI) attempts to solve the limitations of the modular stack by collapsing it. The π series of models (π_0, $\pi_{0.6}$, $\pi^*_{0.6}$) are Vision-Language-Action (VLA) models. This architecture posits that perception, reasoning, planning, and control are not separate problems, but rather a single translation task: translating pixels and text into motion.

2.1 The Backbone: Scaling Internet Knowledge to the Physical World

The foundation of the $\pi_{0.6}$ model is a pre-trained Vision-Language Model (VLM). The specific architecture utilizes a 5-billion parameter backbone, likely initialized from models such as PaliGemma or SigLIP combined with a Gemma decoder.[1] This choice is strategic. By starting with a model pre-trained on internet-scale text and images, the robot inherits a vast semantic understanding of the world before it ever sees a robot arm. It knows what a "spoon" looks like, it understands the concept of "folding," and it grasps the physics of "liquids" in a static, semantic sense.

However, a standard VLM outputs text tokens. A robot cannot act on the token "move_arm." It requires continuous values: joint positions (radians), velocities, and gripper widths. To bridge this gap, Physical Intelligence augments the backbone with a specialized component: the **Action Expert**.

2.1.1 The Action Expert and Architecture splitting

The $\pi_{0.6}$ architecture introduces a bifurcation in the transformer's processing. The main backbone processes the visual and textual inputs to build a rich latent representation of the scene. This representation is then fed into the Action Expert—a dedicated module comprising approximately 860 million parameters.[7]

This design serves two purposes:

1. **Specialization:** The Action Expert focuses solely on the high-frequency dynamics of motor control, learning the precise kinematics of specific robots.
2. **Insulation:** It prevents the catastrophic forgetting of semantic knowledge. If one were to fine-tune the entire 5B parameter model on robot data, the model might forget what a "toaster" is while learning to grasp it. By insulating the backbone and training the Action Expert (and potentially lightweight adapters), the system retains its general-world

knowledge while acquiring physical proficiency.[1]

2.2 Perception and Planning "In One Shot"

The user's query specifically asks how perception and planning are done "in one shot." This is the defining characteristic of the VLA paradigm. In the $\pi_{0.6}$ architecture, there is no A* search, no RRT tree, and no separate occupancy grid.

2.2.1 The Collapse of the Loop

When the $\pi_{0.6}$ model receives an image of a messy table and the instruction "clean the trash," it does not explicitly map the table. It does not calculate coordinates for the trash. Instead, the perception-action loop is collapsed into a single forward pass of the neural network.

1. **Tokenization:** The image is broken into patches and tokenized. The text instruction is tokenized. The robot's current proprioceptive state (joint angles) is tokenized.
2. **Attention:** The Transformer layers perform massive self-attention operations. The model attends to the "trash" pixels because the text token "trash" activates those specific visual features in the high-dimensional latent space.
3. **Generation:** The Action Expert decodes this latent state directly into a sequence of actions.

2.2.2 Planning as Conditional Generation

In this paradigm, "planning" is a generative process. Just as GPT-4 "plans" a sentence by predicting the most probable next word based on the context, $\pi_{0.6}$ "plans" a motion by predicting the most probable trajectory based on the visual context. The "plan" is implicit. The model has seen thousands of examples of trash being picked up. It has learned the statistical correlation between the visual appearance of trash, the robot's current arm position, and the sequence of motor commands required to connect the two.

This is often referred to as "System 1" robotics—intuitive, fast, and reactive. The robot "hallucinates" the correct trajectory based on learned priors. This is the "One Shot": a single inference step (which may involve multiple internal denoising steps, detailed below) replaces the iterative loop of perception, estimation, and search.

2.3 The Mechanism of Action: Flow Matching

A critical innovation in the π architecture is how it represents these continuous actions. Standard regression (predicting the mean action) fails in robotics because valid actions are often multi-modal. (e.g., To go around a pole, you can go left or right. The average of left and right is straight into the pole). Diffusion models solve this but are often slow and jittery.

Physical Intelligence employs **Flow Matching**, a newer class of generative models.[1]

- **Concept:** Flow matching learns a Vector Field that transforms a simple probability distribution (like Gaussian noise) into the complex distribution of valid robot trajectories.
- **Advantage:** Unlike diffusion, which models a stochastic (random) path to the solution, Flow Matching models a deterministic Ordinary Differential Equation (ODE). This results in straighter trajectories in the probability space, which allows for faster sampling and more temporally consistent robot motions.
- **The Output:** The model does not output a single step. It outputs an **Action Chunk**—a sequence of actions (typically 50 steps, representing 1 second of motion).[8] This chunk represents the "local plan." By predicting 1 second into the future, the model ensures that the immediate action is part of a coherent, smooth trajectory, effectively performing short-horizon planning at every inference step.

2.4 The Evolution to $\pi^*_{0.6}$: The RECAP Method

The base $\pi_{0.6}$ model is trained via Imitation Learning (Behavior Cloning). While effective, imitation is limited by the quality of the demonstrator and suffers from "covariate shift" (if the robot drifts off the path, it doesn't know how to recover because it never saw errors in the training data).

To address this, Physical Intelligence introduced $\pi^*_{0.6}$, trained using the **RECAP** method (RL with Experience & Corrections via Advantage-conditioned Policies).[2] This reintroduces a form of "System 2" optimization into the learning process.

2.4.1 The Three-Stage Learning Process

1. **Demonstration (Imitation):** The model is pre-trained on expert data (the standard $\pi_{0.6}$ phase).
2. **Correction (Intervention):** Human operators watch the robot perform tasks. When the robot fails, the human takes over and corrects the behavior. This provides the model with crucial data on "how to recover from failure," solving the covariate shift problem.
3. **Autonomous Practice (Reinforcement Learning):** The robot practices the task autonomously. A Value Function (Critic) estimates the quality of the robot's actions. The Policy (Actor) is fine-tuned to maximize this value.

2.4.2 Advantage Conditioning

A key technical detail of $\pi^*_{0.6}$ is "Advantage Conditioning." During training, the model is fed a token representing the "Advantage" (how much better the current action is compared to the average). During inference, the operator "prompts" the robot with a high-advantage token. Effectively, this commands the robot: "Execute this task as if you were in the 99th percentile of your best performances." This RL-finetuning stage doubled the throughput on complex tasks like espresso making and laundry folding.[2]

---

3. Inputs, Frequency, and The "Missing" Lidar

To understand the operational envelope of the $\pi_{0.6}$ network, we must scrutinize its I/O profile. The user specifically asked about inputs and frequency.

3.1 Sensory Inputs: The Vision-Proprioception Pair

The $\pi$ architecture is decidedly vision-centric. The primary inputs tokenized by the VLA are:

Input Modality	Details & Specifications	Source
Global Vision	RGB images from a base-mounted or external "third-person" camera. This provides scene context. Resolution is typically 448×448 px.	[1]
Local Vision	RGB images from wrist-mounted cameras (one per arm). These provide high-fidelity views of the interaction zone, critical for fine manipulation (e.g., inserting a portafilter).	[1]
Proprioception	A vector q_t containing joint positions (angles in radians), joint velocities, and gripper states (width/force). This vector is projected via a linear layer into the transformer's embedding dimension.	[1]
Language	Natural language text instructions.	[1]
Conditioning	Latent vectors or scalar tokens representing "Advantage" (in π^*) or other task metadata.	[7]

3.2 The Absence of Lidar and Explicit Navigation Stacks

A thorough review of the $\pi_{0.6}$ technical documentation reveals a significant omission:

There is no mention of Lidar point clouds or IMU streams being tokenized and fed into the VLA backbone..[1]

This is a profound divergence from the industry standard. In a ROS-based mobile manipulator (like the Fibocom or Trossen robots used by PI), the navigation stack almost always relies on a 2D Lidar to build an occupancy grid for SLAM.

- **The PI Approach:** By omitting Lidar from the VLA, Physical Intelligence is adopting a **Visual Navigation** strategy. The VLA controls the mobile base (x, y, θ velocity) based solely on visual inputs. The model must learn to infer depth, obstacle distance, and drivable terrain from the RGB images alone.
- **Implications:** This simplifies the hardware sensor suite and unifies the "brain" (one model for driving and grasping). However, it places an immense burden on the perception network to be robust to lighting changes, textureless surfaces, and mirrors—conditions where Lidar typically excels.

3.3 Frequency and Real-Time Chunking (RTC)

Robots live in continuous time. A major challenge for foundation models is **Inference Latency**. Running a 5-billion parameter model takes time—often 100ms to 300ms even on high-end H100 GPUs.

- **The Problem:** If a robot needs to update its motor commands at 50 Hz (every 20ms), but the model takes 200ms to "think," the robot would have to stop and wait 10 cycles between every move. This leads to stuttering, "stop-and-go" behavior.

3.3.1 The Solution: Real-Time Chunking (RTC)

Physical Intelligence utilizes a technique called Real-Time Chunking (RTC) to solve this.[8]

1. **Prediction:** At time t, the model predicts a "chunk" of actions for the next second (t to $t+50$).
2. **Execution:** The robot begins executing this chunk.
3. **Pipelining:** *While* the robot is executing the current chunk, the model captures the next image and begins computing the *next* chunk.
4. **Stitching:** By the time the robot nears the end of the current chunk (say, at $t+40$), the model has finished computing the next chunk. The system blends the end of the old chunk with the beginning of the new chunk.

This allows the $\pi_{0.6}$ system to maintain a smooth **50 Hz control frequency** despite the heavy inference latency. The system effectively "lives in the future," constantly predicting 1 second ahead to buffer against its own processing delay.

---

4. Dyna Robotics: The Simulation-First Specialist

While Physical Intelligence aims for broad generalization via internet-scale data, Dyna Robotics (Dyna) targets **industrial robustness** via simulation-scale data. Their approach is not to be a "Generalist" immediately, but to be a "Specialist" that works so reliably (99.9%) that it becomes ubiquitous.

4.1 The Technical Lineage: *Eureka* and *DrEureka*

Although Dyna Robotics has not released a whitepaper as detailed as π_0, the company's technical DNA is explicitly traced to the research of its co-founder, Jason Ma, specifically the papers *Eureka* and *DrEureka*.[5] This lineage defines the Dyna architecture.

4.1.1 The Reward Engineering Bottleneck

In Reinforcement Learning (RL), the agent learns by trial and error to maximize a "Reward Function." Defining this function is the hardest part of robotics.

- **Sparse Reward:** "Did you fold the napkin?" (Yes/No). This is too hard to learn; the robot flails for millions of steps without a single "Yes."
- **Dense Reward:** "Distance of hand to napkin corner + smoothness of motion - force applied..." Hand-tuning these coefficients is tedious and often results in "reward hacking" (the robot finds a lazy way to get points without doing the task).

4.1.2 The Dyna Solution: LLM-Written Rewards (*Eureka*)

Dyna's core innovation is automated reward engineering. Instead of a human writing the reward function, an LLM (like GPT-4) writes it.

1. **Input:** The system takes the environment code (simulation) and a plain text task description ("Fold the napkin").
2. **Generation:** The LLM writes executable Python code for a reward function.
3. Evolution: The system trains a policy using this reward code. It measures the success. If the robot fails, the LLM analyzes the failure and "rewrites" the reward code to be better. This "Evolutionary Coding" loop allows Dyna to generate highly complex, effective reward functions that no human could intuit.11

4.2 Sim-to-Real Transfer: *DrEureka*

Training in simulation is fast, but simulated physics never perfectly match the real world (The Sim-to-Real Gap). Dyna bridges this using **Domain Randomization (DR)**.

- **Traditional DR:** A human guesses: "Let's vary the friction by 10% and the mass by 5%."
- **DrEureka:** An LLM analyzes the physics of the task and the simulation policy. It sets the randomization bounds automatically. It might decide that for napkin folding, "stiffness" needs to vary by 50% but "friction" only by 5%.
- **Result:** This produces a policy that is incredibly robust. It has "seen" every possible variation of the napkin's physics during training. When deployed on the real robot (DYNA-1), the real napkin just feels like one of the millions of variations it mastered in the matrix.

4.3 DYNA-1 Performance Metrics

The result of this rigorous Sim-to-Real pipeline is evident in the performance metrics. DYNA-1 demonstrated a **99.4% success rate** in napkin folding over a 24-hour continuous run.[4]

- **Comparison:** Most academic research papers (and even many VLA demos) report success rates in the 80-90% range.
- **The "Nines" Gap:** In industry, the difference between 90% and 99.4% is the difference between a research toy and a product. 90% means the robot fails every 10 minutes (requiring human help). 99.4% means it runs all day. Dyna focuses on closing this gap.

---

5. Comparative Architecture Analysis

The following analysis juxtaposes Physical Intelligence, Dyna Robotics, and the Traditional Modular Stack across key technical dimensions.

5.1 Perception and Planning

Feature	**Traditional Robotics**	**Physical Intelligence (π0.6)**	**Dyna Robotics (DYNA-1)**
Paradigm	**Modular Pipeline** (Map $\to$ Plan $\to$ Act)	**End-to-End Generative** (Context $\to$ Trajectory)	**End-to-End Optimization** (State $\to$ Policy)
Planning	Explicit Search (A*, RRT). Deterministic and geometric.	Implicit Generation. Probabilistic and semantic.	Implicit Policy. Learned reflexes optimized for reward.
Adaptability	Brittle. Fails if geometry is unknown or sensors are noisy.	High Semantic Generalization. Can handle novel objects/instructions.	High Physical Robustness. Can handle messy physics/disturbances.
One-Shot?	No. Sequential processing of distinct modules.	**Yes.** Latent space translation in a single pass.	**Yes.** Policy network forward pass.

5.2 Data Engines and Generalization Sources

- **Physical Intelligence:** Relies on **Cross-Embodiment Data**. By training on data mixed

from UR5e, Franka, and Trossen robots, the π model learns a hardware-agnostic representation of motion. It generalizes because it has seen the "concept" of folding across many different arms.
  - *Strength:* Open-World Generalization (New homes, new objects).[12]
- **Dyna Robotics:** Relies on **Simulation Scale**. By simulating millions of years of contact physics, the DYNA model learns the fundamental dynamics of interaction. It generalizes because it has mastered the *physics* of the task, making it robust to lighting, friction, or cloth texture changes.
  - *Strength:* Closed-World Robustness (Factory reliability, precision).[4]

5.3 Frequency and Latency Management

Feature	Physical Intelligence	Dyna Robotics
Control Freq	50 Hz	50-100 Hz (Typical for RL)
Latency	High (100-300ms). Requires VLM inference.	Low (<20ms). Policy networks are usually smaller (MLP/Small Transformer).
Mitigation	**Real-Time Chunking (RTC)**. "Predict the future" to mask latency.	**Distillation**. Compress the large teacher policy into a small student network.
Compute	Heavy Edge/Cloud (H100 required for training, significant edge compute for inference).	Efficient Edge (Jetson Orin sufficient for inference).

5.4 Comparison Table: Inputs and Sensors

Feature	Physical	Dyna Robotics	Traditional Stack

	Intelligence		
Primary Sensor	RGB Cameras (Base + Wrist)	RGB-D Cameras (Depth is key for Sim-to-Real)	Lidar (2D/3D) + RGB-D
Lidar Usage	**None** (Visual Navigation)	Likely None (Stationary focus)	**Heavy** (SLAM, Safety)
State Input	Proprioception (q_t)	Proprioception + History	Proprioception + IMU
Prompting	Multimodal (Text, Image, State)	Task ID / Goal State	Coordinates / Waypoints

6. Conclusion and Future Outlook

The technical divergence between Physical Intelligence and Dyna Robotics highlights two competing philosophies on the path to Embodied AGI.

Physical Intelligence is building the "System 1" of robotics. The $\pi_{0.6}$ architecture, with its VLM backbone and Flow Matching policy, creates a robot with intuition. It perceives and plans in one shot, translating semantic intent ("clean the kitchen") directly into fluid motion. This architecture is essential for the unstructured chaos of the human home. Its reliance on visual navigation and the exclusion of Lidar signals a bold bet on the sufficiency of vision transformers. However, its current reliance on heavy compute and the probabilistic nature of its generative planning may pose challenges for safety-critical industrial applications.

Dyna Robotics is building the "System 2" capabilities, pre-compiled into a "System 1" reflex. By using LLMs to perform the heavy lifting of reasoning and reward design *offline*, and distilling that into a robust RL policy *online*, Dyna achieves the reliability (99.4%) required for commercial deployment today. It sidesteps the latency and hallucination issues of VLMs by grounding its behavior in rigorous physics simulation.

The Convergence: The future likely lies in the synthesis of these two. We expect to see architectures where a "Physical Intelligence" VLM (acting as the high-level semantic planner) feeds instructions and constraints to a "Dyna" RL policy (acting as the low-level reflexive controller). This hierarchical foundation model would combine the semantic breadth of the internet with the physical mastery of simulation, finally bridging the gap between the robot that understands "fold the shirt" and the robot that can actually do it, ten thousand times in a row, without fail.

For professional roboticists, the implications are clear: the era of writing C++ motion planners is ending. The era of curating datasets, designing simulation benchmarks, and managing foundation model inference is beginning.

References:
.1

Works cited

1. π0: A Vision-Language-Action Flow Model for General Robot Control - Physical Intelligence, accessed November 30, 2025, https://www.physicalintelligence.company/download/pi0.pdf
2. [2511.14759] $π^{*}_{0.6}$: a VLA That Learns From Experience - arXiv, accessed November 30, 2025, https://arxiv.org/abs/2511.14759
3. π 0 : Our First Generalist Policy - Physical Intelligence, accessed November 30, 2025, https://www.physicalintelligence.company/blog/pi0
4. Dyna Robotics unveils 'breakthrough in robust, real-world embodied AI', accessed November 30, 2025, https://roboticsandautomationnews.com/2025/05/01/dyna-robotics-unveils-breakthrough-in-robust-real-world-embodied-ai/90152/
5. Language Model Guided Sim-To-Real Transfer - DrEureka, accessed November 30, 2025, https://eureka-research.github.io/dr-eureka/
6. Comparing 5 Pioneering Robotics Foundation Models for ML-Based Control | by Genki Sano (Co-founder & CTO, Telexistence Inc.) | Medium, accessed November 30, 2025, https://medium.com/@genki-sano/a-practical-comparison-of-five-leading-ml-based-robotics-control-approaches-49e1977dd3ec
7. A VLA that Learns from Experience - Physical Intelligence, accessed November 30, 2025, https://www.physicalintelligence.company/blog/pistar06
8. Real-Time Action Chunking with Large Models - Physical Intelligence, accessed November 30, 2025, https://www.physicalintelligence.company/research/real_time_chunking
9. π_0: A Vision-Language-Action Flow Model for General Robot Control - arXiv, accessed November 30, 2025, https://arxiv.org/html/2410.24164v1
10. Yecheng (Jason) Ma - GRASP Lab, accessed November 30, 2025, https://www.grasp.upenn.edu/people/yecheng-jason-ma/
11. Eureka | Human-Level Reward Design via Coding Large Language Models, accessed November 30, 2025, https://eureka-research.github.io/
12. A VLA with Open-World Generalization - Physical Intelligence, accessed November 30, 2025, https://www.physicalintelligence.company/blog/pi05

13. Physical Intelligence (π), accessed November 30, 2025, https://www.physicalintelligence.company/
14. π0: A Foundation Model for Robotics with Sergey Levine - 719 - YouTube, accessed November 30, 2025, https://www.youtube.com/watch?v=5mY71rGXAkM
15. Vision-language-action model - Wikipedia, accessed November 30, 2025, https://en.wikipedia.org/wiki/Vision-language-action_model
16. [Paper Review] Pi0, Pi0.5, Pi0-FAST - Tracing the Path of Physical Intelligence (PI), accessed November 30, 2025, https://bequiet-log.vercel.app/pi-review
17. DYNA Robotics - Commercial-Grade Robots for Real-World Automation, accessed November 30, 2025, https://www.dyna.co/
18. Dyna Robotics Turbocharges Foundation Model Training with Alluxio, accessed November 30, 2025, https://www.alluxio.io/customer-stories/dyna-robotics
19. The Model's the Thing: Dyna's AI-first Approach to Building Humanoids, accessed November 30, 2025, https://www.automate.org/ai/industry-insights/the-models-the-thing-dynas-ai-first-approach-to-building-humanoids
20. Dyna Robotics Closes $120M Series A: How We Think About Scaling Robotic Foundation Models, accessed November 30, 2025, https://www.dyna.co/blog/dyna-robotics-closes-120m-series-a
21. Research - DYNA Robotics, accessed November 30, 2025, https://www.dyna.co/research
22. Dyna Robotics Raises $120 Million to Advance Robotic Foundation Models on the Path to Physical Artificial General Intelligence - PR Newswire, accessed November 30, 2025, https://www.prnewswire.com/news-releases/dyna-robotics-raises-120-million-to-advance-robotic-foundation-models-on-the-path-to-physical-artificial-general-intelligence-302556817.html
23. Dyna Robotics Unveils DYNA-1: The First Commercial-Ready Robot Foundation Model Offering Fully Autonomous, Round-the-Clock Dexterity | RoboticsTomorrow, accessed November 30, 2025, http://www.roboticstomorrow.com/news/2025/04/30/dyna-robotics-unveils-dyna-1-the-first-commercial-ready-robot-foundation-model-offering-fully-autonomous-round-the-clock-dexterity/24675
24. Physical Intelligence raises $600M to advance robot foundation models, accessed November 30, 2025, https://www.therobotreport.com/physical-intelligence-raises-600m-advance-robot-foundation-models/
25. Welcome, Dyna Robotics! | Salesforce Ventures, accessed November 30, 2025, https://salesforceventures.com/perspectives/welcome-dyna-robotics/
26. Jensen Huang's Investment in a Zhejiang University Alumnus - 36氪, accessed November 30, 2025, https://eu.36kr.com/en/p/3475934730213767
27. Power to the Bot — CRV Partners With Dyna Robotics to Commercialize Embodied AI With Low-Cost… - Medium, accessed November 30, 2025, https://medium.com/crv-insights/power-to-the-bot-crv-partners-with-dyna-rob

otics-to-commercialize-embodied-ai-with-low-cost-af67bebf0318

28. The Robotics Breakout Moment | Salesforce Ventures, accessed November 30, 2025, https://salesforceventures.com/perspectives/the-robotics-breakout-moment/