ChatGPT

Computing Innovation Abstracts (May 2024–May 2025)

Group A: Clear Contracts

Each of the following innovations has a simple, well-defined input/output structure and a narrow purpose – much like a function in code with clear contracts.

States Turn to AI to Spot Guns at Schools

Source: *Associated Press* (May 12, 2024) [1]
Abstract: Several U.S. states are considering or enacting programs to fund **AI-powered surveillance systems** that automatically detect people carrying firearms in school camera feeds [1]. A prominent example is ZeroEyes, a computer-vision system that analyzes live video for visible guns and alerts authorities within seconds. In Kansas, a pending bill was *written so narrowly* (requiring the AI to be patented and already used in over 30 states) that **only ZeroEyes meets the criteria**, raising questions about lobbying [2]. The goal is to enhance school safety by identifying threats faster than human staff could.

Analysis:
- **Category:** Clear (single-purpose AI surveillance)
- **Computing Inputs:** Live security camera video streams in schools.
- **Outputs:** Real-time alerts/notifications when a firearm is detected, including location info.
- **Purpose:** To *quickly identify and respond to gun threats* on school grounds, potentially saving lives by speeding up lockdowns or police intervention.
- **Notes on Student Accessibility:** Students can easily grasp this as a **function that takes images as input and returns a "gun detected" alert as output**. It mirrors familiar systems like face-detection on phones, making it concrete. In class, it's useful for discussing accuracy, false alarms, and ethical implications (e.g. privacy, bias) of AI in security.

Smart Cameras Spot Wildfires Before They Spread

Source: *The Wall Street Journal* (Mar. 2, 2025) [3]
Abstract: The ALERTCalifornia project in California deploys a network of **over 1,150 mountaintop cameras** paired with AI "digital lookouts" to catch wildfires in their earliest stages [3]. The AI system continuously scans live camera feeds from fire-prone areas and has **detected more than 1,200 fires**, sometimes alerting authorities *faster than 911 callers* about one-third of the time [4]. When the AI flags smoke or flame, a human-operated command center verifies the blaze and dispatches firefighters. This early warning system exemplifies a straightforward input-output loop: camera imagery in, fire alert out.

Analysis:
- **Category:** Clear (automated visual monitoring)
- **Computing Inputs:** Streaming video images from remote wildfire cameras (image data of landscapes).

- **Outputs: Alerts/alarms** when smoke or fire signatures are recognized, including the location for firefighters.
- **Purpose:** To *detect wildfires faster* than human spotters or reports, allowing fire crews to respond before small fires explode into large wildfires.
- **Notes on Student Accessibility:** This innovation is easy to frame as a function: **input = camera image, output = boolean "fire detected" plus an alert**. High school students can appreciate the direct impact (preventing disasters) and can discuss how **accuracy** and **false positives** matter. It's also a chance to talk about training AI on images (pattern recognition) in a real-world context that's relevant and engaging.

BCI Decodes Words "Spoken" in the Brain in Real Time

Source: *Medical Xpress* (May 14, 2024) [5]
Abstract: Researchers at Caltech developed a **brain–computer interface (BCI)** that can translate neural signals into words in real time [5]. The system records firing neurons in a speech-related brain region and was trained on a small vocabulary (six specific words plus some control "nonsense" words). In tests with two paralyzed participants, the BCI could identify the words the person *was trying to say in their mind* – with one participant achieving about **79% accuracy** (though performance varied) [6]. This technology essentially takes brain activity as input and produces decoded words as output, demonstrating a clear if extraordinary mapping from thought to computer action.

Analysis:
- **Category:** Clear (single-domain BCI function)
- **Computing Inputs:** Electrical signals from individual neurons in the brain's speech areas (captured via implanted electrodes).
- **Outputs:** Decoded words or text corresponding to the **thought-of speech** (the words the user is attempting to say internally).
- **Purpose:** To *enable communication for people who cannot speak*, by directly translating imagined words into computer-recognized speech or text. (For example, a paralyzed patient could "think" **yes/no** or basic needs and have the system speak it aloud.)
- **Notes on Student Accessibility:** While the neuroscience context is advanced, at its core this is like a function from **brain signals to words** – a concept students find intriguing. It can spark discussion on how training data (limited vocabulary) affects output and the challenges of interpreting noisy inputs. It's also inspirational, showing how computing can restore abilities (paralleling how a function "restores" output from novel input).

Generative AI Scans Your Amazon Packages for Defects Before Shipment

Source: *Fast Company* (June 3, 2024) [7]
Abstract: Amazon has deployed an AI system (code-named *"Project P.I."*) that uses **generative AI and computer vision** to inspect packages in its fulfillment centers [8]. As each box is readied to ship, the AI analyzes images of the package's contents to verify that the correct items are included and undamaged. The model can flag items that are the **wrong size, wrong color, or defective**, preventing packing mistakes [8]. According to Amazon, this not only improves the customer experience but also supports sustainability by reducing waste and reshipments (a nod to using AI to meet climate goals) [9]. The system's function contract is easy to see: input a package image, output a decision or alert if something's amiss.

Analysis:
- **Category:** Clear (quality control via vision AI)
- **Computing Inputs:** Photos or sensor images of each package's contents (and possibly package metadata like expected items).
- **Outputs: Pass/fail signal** for the package inspection – e.g. "okay to ship" vs. "flag for review" (with details like "item missing" or "wrong item detected").
- **Purpose:** To *ensure customers get the correct, intact products*. In practice this means catching packing errors or damaged goods **before** shipment, cutting down on returns and improving efficiency.
- **Notes on Student Accessibility:** Students can relate this to a **real-life if-statement**: "if package is incorrect, flag it." It shows AI in a warehouse setting, which is concrete (especially with the image of a box being laser-scanned). It's a chance to discuss how generative AI might be used in vision tasks and how such a system must be trained to recognize a huge variety of products. The straightforward input-output makes it a great example of a "function" that could be implemented (at a basic level) in a classroom simulation.

3D Printing Paves Way for Personalized Medication

Source: *University of Nottingham News* (May 14, 2024) [10]
Abstract: Scientists at the University of Nottingham have developed a **3D printing technique for pharmaceuticals** that can combine multiple medications into one customized pill [10]. The method, called

Multi-Material Inkjet 3D Printing (MM-IJ3DP), uses special UV-sensitive molecules as "ink." Different drug compounds can be printed in precise micro-structures so that they **release at controlled rates** once ingested [11]. In essence, the printer takes a patient's prescription needs as input and outputs a single poly-pill tailored in dosage and timing – potentially simplifying treatment for patients with complex medication regimens.

Analysis:
- **Category:** Clear (defined manufacturing process)
- **Computing Inputs:** Digital drug composition and dosage specifications for an individual (which drugs, what amounts, and desired release schedule).
- **Outputs:** A **single 3D-printed pill** containing the multiple medications in structured layers or regions (designed to release at different times or in different parts of the gut).
- **Purpose:** To *personalize medicine*, combining what used to be many pills into one and controlling release kinetics. This improves patient compliance (one pill instead of many) and can optimize therapy by timing drug release.
- **Notes on Student Accessibility:** This is a tangible use of computing in health – students can imagine a "medicine printer function." Input parameters (drug A dose, drug B dose, release timing) go into a printer, output is a pill. It's straightforward to discuss how a program might control the printhead to achieve the desired output. While the chemistry is complex, the concept of **tailoring outputs to individual inputs** is a perfect analogy for coding functions. Students might also consider testing and safety (verifying that the output pill meets requirements – a kind of unit test for a printer).

MTA Used Google Pixels to Identify Subway Track Defects

Source: *Engadget* (Feb. 28, 2025) [12]
Abstract: New York City's transit authority (MTA) partnered with Google to pilot an **AI-based track monitoring tool** called *TrackInspect*. In this system, **off-the-shelf smartphones (Google Pixel)** were mounted in subway cars to continuously listen and feel for anomalies as trains ran their routes [13]. The phones' accelerometers, gyroscopes, and microphones gathered vibration and sound data, which machine learning algorithms analyzed to pinpoint sections of track with potential defects or wear [13]. Over a few months, the system successfully identified about **92% of the track problems** that human inspectors later confirmed [14]. The clear contract: raw sensor data goes in, and alerts about track issues come out – allowing targeted maintenance before bigger failures occur.

Analysis:
- **Category:** Clear (sensor data processing)
- **Computing Inputs:** Smartphone sensor readings (vibrations, noise, magnetic signals) collected on moving trains.
- **Outputs: Detected track defect reports** – essentially flags for specific locations that likely have an issue (e.g. rail crack, misalignment), often with a confidence score or priority level.
- **Purpose:** To *perform preventative maintenance on subway infrastructure*. By automating detection of track problems, the system helps the MTA fix issues proactively, improving safety and reducing delays from unexpected breakdowns.
- **Notes on Student Accessibility:** This example connects to everyday technology (smartphone sensors) used in a clever way. Students can see it as a function that takes a **time series of acceleration values** and outputs whether there's an anomaly. It's great for discussing how we can treat real-world analog signals as

inputs to a computational problem. The 92% success statistic also invites conversation about false negatives and the importance of verification – just like checking if a function's output is correct.

Group B: Ambiguous or Layered Contracts

These innovations involve **multi-layered inputs, context-dependent outputs, or complex/ethical considerations**. They behave less like a simple one-to-one function and more like layered systems, making their "function contracts" fuzzier or situational.

New Techniques to Stop Audio Deepfakes

Source: *IEEE Spectrum* (May 30, 2024) [15]
Abstract: In a U.S. contest to combat AI-generated voice **"deepfakes,"** researchers showcased *multiple complementary innovations*. The winning entries included **OriginStory** – a modified microphone that monitors a speaker's physiological signals (like throat vibrations) to verify a live human voice – and **AI Detect** – software that embeds machine learning into audio processing devices to **spot signs of an artificial voice** in real time [15]. Another solution, **DeFake**, adds subtle distortions to genuine recordings to *prevent* adversaries from cloning a voice [16]. These layered defenses illustrate that combating deepfakes often requires *stacking inputs* (biosignals + audio data + signal processing) and outputs that range from authentication decisions to preemptive signal tweaks. The "contract" here is broader and context-dependent compared to a single detection function.

Analysis:
- **Category:** Ambiguous/Layered (multi-input authentication & prevention system)
- **Computing Inputs:** Various signals *at once* – e.g. the audio waveform of speech, plus biometric readings (vocal cord vibrations, etc.) for OriginStory; or just audio but analyzed with different models as in AI Detect; and even an original audio file in DeFake that gets an added watermark noise.
- **Outputs: Multi-faceted outcomes**: a confidence score or boolean indicating a voice is real vs. fake (for detection tools), and in the case of DeFake, an output audio file that remains intelligible to humans but resists AI cloning.
- **Purpose:** To *defend against AI voice spoofing*, protecting security (like voice authentication systems) and trust in media. The purpose spans **detection** (catch a fake) and **prevention** (make audio hard to fake) – a complex, layered goal.
- **Notes on Student Accessibility:** Students can discuss why a simple function `isFake(voice)` isn't enough here. Each approach deals with the problem differently (hardware vs. software vs. signal alteration), showing that complex problems require **composed solutions**. This example is great for ethics and security discussions – e.g., trade-offs of adding noise to recordings, or privacy issues in continuously monitoring biometrics. It highlights that inputs/outputs can be numerous and that defining a "contract" sometimes means specifying several sub-functions working together.

Self-Replicating 'Life' Created from Digital 'Primordial Soup'

Source: *New Scientist* (July 9, 2024) [17]
Abstract: Google researchers experimentally produced a form of **artificial life inside a computer** without explicitly programming it in. They dumped tens of thousands of small snippets of code into a shared environment – a "digital primordial soup" – and allowed them to randomly combine, mutate, and execute

[18]. Over time, some of these code aggregates **began to self-replicate**, copying themselves and competing for resources (memory/processing time) until they reached a population cap [19]. New types of replicators even emerged and outcompeted the older ones [19]. This outcome (evolving digital organisms) wasn't a predefined output at all – it arose from many stochastic interactions. The *inputs* and *outputs* here are ambiguous: the system's initial state was random code, and the "output" was an emergent behavior (self-replication) that the researchers observed rather than directly programmed.

Analysis:
- **Category:** Ambiguous/Layered (emergent behavior from open-ended system)
- **Computing Inputs:** Initially, **random fragments of code** and a set of rules for how they can interact (in this case, allowing recombination and execution with no specific goal). There is no single input after initialization; the system is more like a Petri dish evolving on its own.
- **Outputs: Emergent code behaviors**, especially self-replicating programs. There isn't a single numeric or categorical output; instead, the outcome is a complex dynamic: certain programs multiply, others die out. One could say the "output" was the set of surviving self-replicators and their activity.
- **Purpose:** To *explore the origin of life and evolution in a computational context*. The researchers aimed to see if lifelike behaviors (like reproduction and evolution) could spontaneously arise from randomness + selection, which might give insights into biology and AI.
- **Notes on Student Accessibility:** This is a fascinating case where there's no simple input→output as in a normal function. Students can discuss how **unpredictability** and **emergence** play a role. It's an opportunity to contrast deterministic programs with evolutionary algorithms or simulations. In a classroom, one might simulate a simpler "digital ecosystem" – but the key lesson is that not all programs have a clear contract; some are **open-ended**, producing surprising results (which raises questions like how to test or verify such systems).

Stores Roll Out AI-Powered Vending Machines That Sell Bullets

Source: *Gizmodo* (July 5, 2024) [20]
Abstract: An Oklahoma-based company, American Rounds, is installing **vending machines for ammunition** in convenience stores – an application with layered AI-driven safety checks. To buy bullets, a customer must scan their ID and face; the machine's AI then uses **image recognition** to match the face with the ID and verify the buyer's age and identity [20]. Only if all checks pass will the machine unlock and dispense the ammunition. The system integrates multiple inputs (government-issued ID data, a facial scan, possibly transaction details) and outputs a decision (vend or no-vend) with serious real-world consequences. Unlike a simple "unlock if face matches" phone feature, this involves legal context (age laws) and raises ethical flags – making the "function contract" more complex than it first appears.

Analysis:
- **Category:** Ambiguous/Layered (multi-factor automated system with ethical context)
- **Computing Inputs: ID scan data** (e.g. driver's license info), the buyer's live **camera image** for facial recognition, and potentially database checks (e.g., verifying the ID's validity).
- **Outputs:** A **physical action** (dispensing a box of ammunition) *only if* the person is validated. In addition, there may be logging outputs (record of the sale) or an alert if something fails (e.g., "ID mismatch – sale blocked").
- **Purpose:** To *sell ammunition in a controlled yet automated way*. From the company's perspective, it's convenience (24/7 sales) combined with enforcing gun laws (no underage sales, ensuring the buyer is the ID owner). It's essentially automating a function that a human clerk would do – but without human

judgment if something seems off.
- **Notes on Student Accessibility:** This example can prompt debate in class. Technically, it's like a compound function: **check ID age && match face -> if true, vend**. Students can map that logic easily. However, the **ethical dimension** (should we automate this at all?) and potential failure modes (face recognition errors, fake IDs) highlight how real-world "functions" can have far-reaching impact. It's useful for teaching that a program's "contract" must sometimes include safety or ethical conditions that go beyond the code (e.g., what if the system misidentifies someone – who is accountable?).

'Robotability Score' Ranks NYC Streets for Robot Deployment

Source: *Cornell Chronicle* (Apr. 30, 2025) [21]
Abstract: To guide future urban robot deployments (like delivery robots or sidewalk drones), researchers at Cornell University devised a **"robotability score"** that quantifies how hospitable a city street is for robots [21]. Calculating the score is a multilayered process: they pull in New York City's open data (e.g. sidewalk width, curb ramps, locations of bus stops, bike lanes, newsstands) and combine it with analysis of **8 million street images** to gauge pedestrian and vehicle traffic levels [22]. These diverse inputs feed into a model that outputs a score map – essentially rating each block on how easily a robot could navigate. Because the score depends on many context factors (infrastructure, traffic, etc.), the "function" here is more complex than a typical algorithm: it merges urban planning data, computer vision, and robotics considerations into one result.

Analysis:
- **Category:** Ambiguous/Layered (multi-criteria data fusion for decision support)
- **Computing Inputs: Structured urban data** (street metrics like widths, counts of obstacles, etc.) + **unstructured data** from images (crowd density, vehicle presence extracted via AI vision). Possibly also temporal data (time of day patterns).
- **Outputs:** A **numeric score or rating** per street segment (and perhaps a color-coded map) indicating robot-friendliness. It might rank streets from most robot-navigable to least, and could include multiple scores (e.g., one for sidewalk robots vs. one for road autonomous vehicles).
- **Purpose:** To *inform city planners and companies* where robots can be deployed successfully and what infrastructure improvements might be needed. For example, a low score might highlight where sidewalks are too narrow or congested, suggesting changes before rolling out delivery robots.
- **Notes on Student Accessibility:** This project shows how a "function" can ingest **big data from many sources**. Students can discuss how one might normalize different inputs (how to compare sidewalk width vs. image-derived crowd density quantitatively?). It's a chance to introduce the idea of **weighted factors** in an algorithm. The societal context (robots in public spaces) also invites discussion on accessibility and equity – making it a rich example for not just CS, but also civics/ethics. The concept of a score is simple, but how it's computed is more complex – a good illustration of abstraction (the end-user sees one score, but under the hood many calculations happened).

Eye-Scanning ID Project Launches in U.S.

Source: *CNBC* (Apr. 30, 2025) [23]
Abstract: Worldcoin's "World ID" project – an ambitious digital identity system – opened its first retail locations in the U.S. where people can get their **irises scanned** to create a unique personal ID secured on a blockchain [23]. At these "Orb" scanning stations, a biometric device takes an image of your eyeballs and generates an **IrisCode**, a unique hash that proves you're a real individual (and not a bot or duplicate). The

purpose is to establish a *global* identity verification network: for instance, after scanning, users can log into services like Minecraft or Reddit by proving their World ID, without sharing personal details [24]. The system's inputs and outputs are layered – a human biometric is input, a cryptographic token is output – and its success depends on context like user trust, widespread adoption, and ethical handling of biometric data.

Analysis:
- **Category:** Ambiguous/Layered (biometric blockchain identity system)
- **Computing Inputs: High-resolution iris images** (and associated personal sign-up data) captured by the Worldcoin Orb. These are processed into biometric templates. Also, the system takes in queries from websites/apps asking "Is this user a real unique person?"
- **Outputs:** The primary output is a **cryptographic World ID** (an IrisCode and associated digital identity record). For verification queries, the output is a **yes/no proof of personhood** (e.g., "user X is verified unique human #12345") without revealing the user's actual identity.
- **Purpose:** To *provide a secure, privacy-preserving digital identity* that can't be easily forged or duplicated. In essence, it aims to solve online problems like bots, fake accounts, and identity fraud by offering a **universal login that verifies humanness**.
- **Notes on Student Accessibility:** This innovation combines several deep topics – biometrics, cryptography, AI, privacy. Students might find the Orb device interesting (it looks like a sci-fi silver sphere). As a function, it's not as straightforward as others: you input a physical characteristic and get an abstract digital credential. This can lead to discussions about **one-way functions** (the iris image isn't stored, just a hash) and the **ethics**: Would you scan your eye for a global ID? It's a prime example of tech with societal implications, encouraging students to think about what "output" means (is it the hash, or the societal outcome of having such an ID system?).

Wearable Pediatric Soft Exoskeleton Made of Smart Materials

Source: *University of Houston News* (Apr. 29, 2025) [25]
Abstract: Engineers and physicians in Houston have created **MyoStep**, a soft, wearable exoskeleton designed to help children with cerebral palsy walk more easily [25]. Unlike rigid exosuits, this device uses **smart fabrics and artificial "muscles"** (flexible actuators) along with a network of wireless sensors. The

sensors collect real-time data on the child's movement – for example, detecting when a leg is about to take a step – and the system responds by contracting the artificial muscles at the right moment to assist movement [25]. MyoStep is also adjustable as a child grows. The input (various sensor signals about body position and motion) is complex, and the output (physical force applied in sync with the child's gait) must be precisely timed and tuned to the individual. This makes the technology's function contract layered and adaptive, far beyond a simple input→output code routine.

Analysis:
- **Category:** Ambiguous/Layered (integrated human-device system with feedback loop)
- **Computing Inputs: Motion sensor data** from the child's body (e.g., accelerometers/gyros on limbs, pressure sensors in the exoskeleton straps). Possibly biometric feedback too (muscle activation signals, etc.). All these inputs feed into a control algorithm.
- **Outputs: Mechanical assistive force** applied by the exoskeleton's artificial muscles, timed to the child's steps. There may also be secondary outputs like logs of the child's walking patterns for therapists. The output is essentially a *continuous, context-aware adjustment* rather than a single value – the device constantly decides how much force to add and when.
- **Purpose:** To *improve mobility for children with movement disorders*. Specifically, it helps kids with cerebral palsy build strength and walk with better form by providing support only when needed, encouraging the muscle use when possible and stepping in when necessary. Over time, this can promote better walking ability and physical therapy progress.
- **Notes on Student Accessibility:** This example shows computing embedded in a **physical, real-time system**. It's a nice contrast to screen-based or purely data-based programs. Students can discuss how the "function" here is actually a **control system** – inputs continuously stream in, and the program must continuously generate outputs (forces) in response. It's a great way to introduce concepts like feedback loops. The humanitarian aspect (helping kids) also resonates, and we can ask students to consider how they'd test such a system safely. It demonstrates that not all computing problems output a number or classification; some output *actions in the world*, blurring the line between software and human context.

Digital Twins Used to Improve Built Environments for Robots

Source: *The Engineer (UK)* (June 3, 2024) [26]
Abstract: Researchers in Singapore are leveraging **digital twin technology** to make buildings more robot-friendly. Their approach involves a **three-phase process** [26]: (1) collect detailed data from the real environment (on-site scans, photos, measurements of a building's layout), (2) convert that data into a digital 3D model of the space (a virtual replica of corridors, rooms, obstacles), and (3) use that virtual environment in simulation to test how robots navigate and interact there [26]. By observing robot behavior in the digital twin, designers can identify problem areas (like a doorway too tight for a delivery robot or a sensor blind spot) and then adjust the real-world plans. The inputs include both physical world data and design parameters; outputs might be recommendations or modifications to the environment. The "function" here is multi-layered: it's part reality capture, part simulation, and part analysis – all to ensure robots and buildings work together smoothly.

Analysis:
- **Category:** Ambiguous/Layered (simulation loop with real-world feedback)
- **Computing Inputs: Spatial data** from buildings (point clouds from 3D scanners, architectural drawings), which get translated into a virtual model. Also, **robot specs and behaviors** are inputs to the simulation (e.g., robot dimensions, movement algorithms).

- **Outputs: Insights and design tweaks** for the built environment – for example, “add a ramp here” or “the robot can’t see around this corner, consider a mirror or a sensor.” In the simulation step, outputs could also include performance metrics (like how fast a robot can complete a route in the current design, or where it consistently fails). The final output is often a set of recommendations or validations for architects and engineers.
- **Purpose:** To *ensure that new buildings or spaces are optimized for robotic assistants* (like hospital delivery bots or warehouse AGVs). By catching design issues in a virtual replica, it’s cheaper and safer than discovering them after construction. The ultimate goal is **human-robot coexistence in shared spaces** with minimal friction.
- **Notes on Student Accessibility:** This case highlights how computing bridges to civil engineering and robotics. It’s less about coding a simple function and more about a **pipeline** of processes (scan -> model -> simulate -> advise). Students can learn about digital twins – essentially **virtual sandbox versions of real systems**. This can lead to a discussion of decomposition: one can imagine each phase as its own function (input scan data -> output model, etc.). It shows that sometimes to define a “contract,” you must consider multiple stages and the context in which the output will be used (here, by architects). It’s a good example of how complex real-world tasks can be broken into simpler functions that feed into one another.

---

[1] [2] [5] [6] [10] [11] ACM TechNews
http://technews.acm.org/archives.cfm?fo=2024-05-may/may-17-2024.html

[3] [4] [12] [13] [14] ACM TechNews
http://technews.acm.org/archives.cfm?fo=2025-03-mar/mar-05-2025.html

[7] [8] [9] [15] [16] [26] ACM TechNews
http://technews.acm.org/archives.cfm?fo=2024-06-jun/jun-05-2024.html

[17] [18] [19] [20] ACM TechNews
http://technews.acm.org/archives.cfm?fo=2024-07-jul/jul-10-2024.html

[21] [22] [23] [24] [25] ACM TechNews
http://technews.acm.org/archives.cfm?fo=2025-05-may/may-05-2025.html