IRSTI 20.23.27
https://doi.org/10.56132/2791-3368-2026-2-66-180-203

A.Karim, Y.A. Buirash, N. Remer
Qazaq Youth Science Hub LLP, Almaty, Kazakhstan
E-mail: ersultanbujras@gmail.com

DEVELOPMENT OF AN INTELLIGENT ACCESS CONTROL SYSTEM FOR CRITICAL INFRASTRUCTURE WITH PRIORITY ACCESS FOR EMERGENCY VEHICLES

Abstract. This vehicle recognition access control device and Automatic Number Plate Recognition (ANPR) barrier uses autonomous operation validation for successful anticipated function within such an innovative prototype for future smart city and critical infrastructure developments, where access control is preferred as a decentralized effort without non-centralized solutions.

Therefore, the vehicle recognition access control device operates through access control and novel functionality through ANPR (automatic number plate recognition) and embedded actuation through a local Raspberry Pi and ESP32 webservers, a combination of computer vision and IoT-driven device that runs both hardware and software on the same device.

The proposed system can be applied at military facilities, critical infrastructure sites, healthcare institutions, and other restricted-access areas where rapid and reliable vehicle identification is required. Its decentralized architecture enhances operational resilience and ensures reliable operation under limited network connectivity.

Finally, testing confirmed anticipated operational levels as it confirmed that string-level recognition functionality was successfully recognized at a 94.6% accuracy. However, since one license plate recognized was considered one time recognized for one subject, the average latency of speed of recognition was 73 seconds.

In addition, functional intention received anticipated findings based on real-world situations, operating 1 second after the recognition time. Secondary testing confirmed that the device had been unplugged (for a time) from the internet.

The input and output secured themselves in position at the designated entry point. This supports intentions for stability in decentralized function.

Ultimately, these findings support the feasibility of the developed prototype for smart city applications, military facilities, and critical infrastructure environments where reliable decentralized access control is required.

Keywords: Automatic Number Plate Recognition (ANPR); Intelligent Barrier System; Edge Computing; IoT-Based Access Control; Embedded Systems; Critical Infrastructure Security; Military Facilities; Autonomous Access Control; Vehicle Identification; Emergency Vehicle Priority.

Introduction.

Automated vehicle access control, basically supported and implemented by Automatic Number Plate Recognition (ANPR) systems, has become an integral part of the management of vehicular flows, accelerating throughput, enhancing safety, and reducing human labor.

Yet a crucial open research issue involves emergency vehicle preemption and dependable embedded/edge technology that operates autonomously with limited processing power and inconsistent connectivity [1, 2].

Such an issue is relevant to Kazakhstan since the MES publically directs the population on the matter. In fact, barriers often impede fire engines and ambulances [3], contrary to approved local regulation. The preliminary collection of fire safety documents identifies the prohibition of barriers that prevent access of vehicles used for emergency situations [3] and related MES regulations, while the National Police have established standards for number plates (e.g. Order No. 1040, 19.12.2015 [4]; ST RK 986-2012 [5]). In addition, international codes state that automated gates must possess emergency release options (International Fire Code §503.6 [6]; UL 325 [7]), yet there is no universal standard in Kazakhstan yet implemented. Thus, there is a need for a prototype of an automated universal solution which will meet national and international requirements.

In addition to civilian applications, the problem is particularly relevant for military facilities, healthcare institutions, critical infrastructure sites, and other protected areas where secure access control must be combined with rapid authorization of emergency vehicles. In such environments, uninterrupted operation, local data processing, and independence from centralized communication infrastructure are critical factors affecting the effectiveness and resilience of access control systems. Therefore, the development of intelligent decentralized solutions based on automatic vehicle identification technologies represents an important research and practical task.

Multiple contextual elements exacerbate the timeliness of this problem. Socially, medically, the sooner a responder to an emergency event can be processed, the more successful interventions can be; the more lives can be spared. Economically, automatic access solutions help minimize staffing costs, reduce queue buildup at entry and exit locations and speed up processing times for industrial and residential buildings [8,9]. On a technological level, countless affordable single-board computers (Raspberry Pi), versatile, powerful open-source computer-vision toolkits (OpenCV) and OCR (Tesseract OCR) provide avenues for the development of lightweight local systems with image processing, embedded actuators, and on-board information retention [10, 11, 12]. Furthermore, these edge systems can operate independent of the cloud and compliant with internationally recognized standards such as ISO/TR 6026 and ISO/TR 25221 (ISO, 2013; ISO, 2015).

Yet despite the potential, research is fragmented. For example, much of the existing body of work explores ANPR algorithms [13], concepts for traffic-management systems with emergency responders [14] or gate automation with

simplistic implementations and just embedded computers [13, 15]. The topic of emergency vehicle prioritization is rarely discussed with a comprehensive end-to-end approach for machine learning training, single-board computers as embedded systems, a local server approach, and industrial-complex PCB based development tested in parallel.

Subsequent studies evolved the intelligent vehicle access control systems into smart vehicle access control systems through computer vision, embedded control and IoT communication frameworks. For example, Ullah et al. [16] created a barrier control system with ultrasonic sensors and an image processing, license plate recognition approach for access via Raspberry Pi and detection accuracy. Furthermore, Islam et al. [17] validated the real-time vehicle access system through gradient-based feature extraction and image processing. Fadlianda et al. [18] used an IoT-based automatic gate access system via ESP32 with WiFi for access and RFID as well as ultrasonic obstacle detection. Al Mamun et al. [19] established an IoT-based smart car parking system via integrated sensors and mobile apps for detected spots communicated through MQTT. The latest studies as of this writing focus on implementation, low-power embedded solutions, modularized communication and on-site independence. Leng et al. [20] implemented a low-power edge-computing system via entry-level camera systems at the edge for license plate recognition and real-time application. D'Ortona et al. [21] constructed an open source IoT architecture based on MQTT for decentralized access in smart city architecture. Therefore, the direction of this research shows a more affordable, decentralized parking and access systems that rely on enhanced interoperability and resiliency to network failure and decreased reliance on cloud-computing applications.

This study addresses this gap through the design, implementation, and evaluation of an intelligent ANPR-based access control system incorporating a local server architecture, custom PCB hardware, embedded control logic, and priority access functionality for emergency vehicles.

Materials and Methods.

Research Design.

The research project involves an experimental research design supplemented by prototyping. An experimental research design is implemented because this is not purely an endeavor in theoretical modeling, it's a real-world hardware-software prototype barrier control device. With an experimental research design, the entire control system can be characterized from start to finish; from license plate detection to physical barrier actuation and therefore, software effectiveness and hardware reactivity can be assessed in near-functionality settings.

In addition, beyond the prototype testing and assessment, research is based on a physics-based electromechanical model of the actuator, derived from DC motor operations of a known set of equations of motion, electrical and mechanical (Cunico, Cenedese, Zaccarian, & Borgo, 2022) [22].

$$V = Ri + Ldtdi + Ke\omega. \quad (1)$$

$$Jdtd\omega = Kti - \tau L(\theta,\theta^{\cdot}). \quad (2)$$

Where, V - applied voltage, i - armature current, R - armature resistance, L - armature inductance, Ke - back-EMF constant, ω - angular velocity of the rotor/ J - total moment of inertia of the rotor and load, Kt - torque constant, τL(θ,θ˙) - combined load torque including spring, damping and gravitational effects, θ - angular position of the rotor.

The modeling component enables one to predict system dynamics in a formulaic manner and facilitates controller creation and subsequent simulation before a tangible version is constructed.

Therefore, through theoretical modeling and hardware testing, this study validates the intelligent control system in question on a theoretical (via model based simulation), functional (via software-hardware integration), and empirical (via simulation and prototyping tested to create real world implications).

System Components and Setup.

The system is divided into a software/hardware subsystem that operates as implemented based on the control levels found during the automated barrier opening. The hardware subsystem consists of a Full HD IP camera, Raspberry Pi with which the camera connects, and an ESP32-based PCB with which the ESP32-based relay control board connects motors via relay activation, and motors connect within distance through joined ports.

Sensor-based, cost-effective Raspberry Pi and ESP32 solutions were used by Ullah et al. [16] and Fadlianda et al. [18] in similarly embedded systems, but the following system expands upon these findings as a PCB with additional motors and a known emergency use/access line was implemented for local regulations for emergency-response concerns.

The software subsystem encompasses image taking, plate recognition, and database recognition. The IP camera constantly takes pictures of any vehicle coming towards the barrier and simultaneously sends a signal to the Raspberry Pi, where OpenCV opens access to Tesseract Optical Character Recognition (OCR) for plate recognition of the vehicle, which sends back to the Raspberry Pi a subsequent signal in a respective string of numbers and letters.

This string connects to a local Excel database, which consists of a tab with cars currently given access and an additional tab with vehicles unknown, and a third being dedicated to any emergency vehicle-response registrations. If the string is found to match a registration previously known on the list or the emergency-response tab, a universal asynchronous receiver-transmitter (UART)/wireless signal is sent from the Raspberry Pi back to the ESP32 control board which uses its own code to access a dedicated 24V relay module to which one of two ports was sent via wire connection which connects to the motors of the physical DC Motor/Servo mechanics which appropriately opens the barrier.

All system code was run through Raspberry Pi through Python with FLASK as a microframework for localized use without reliance on the internet or additional databases, while additional Arduino C++ Firmware was added for immediate action and stabilization.

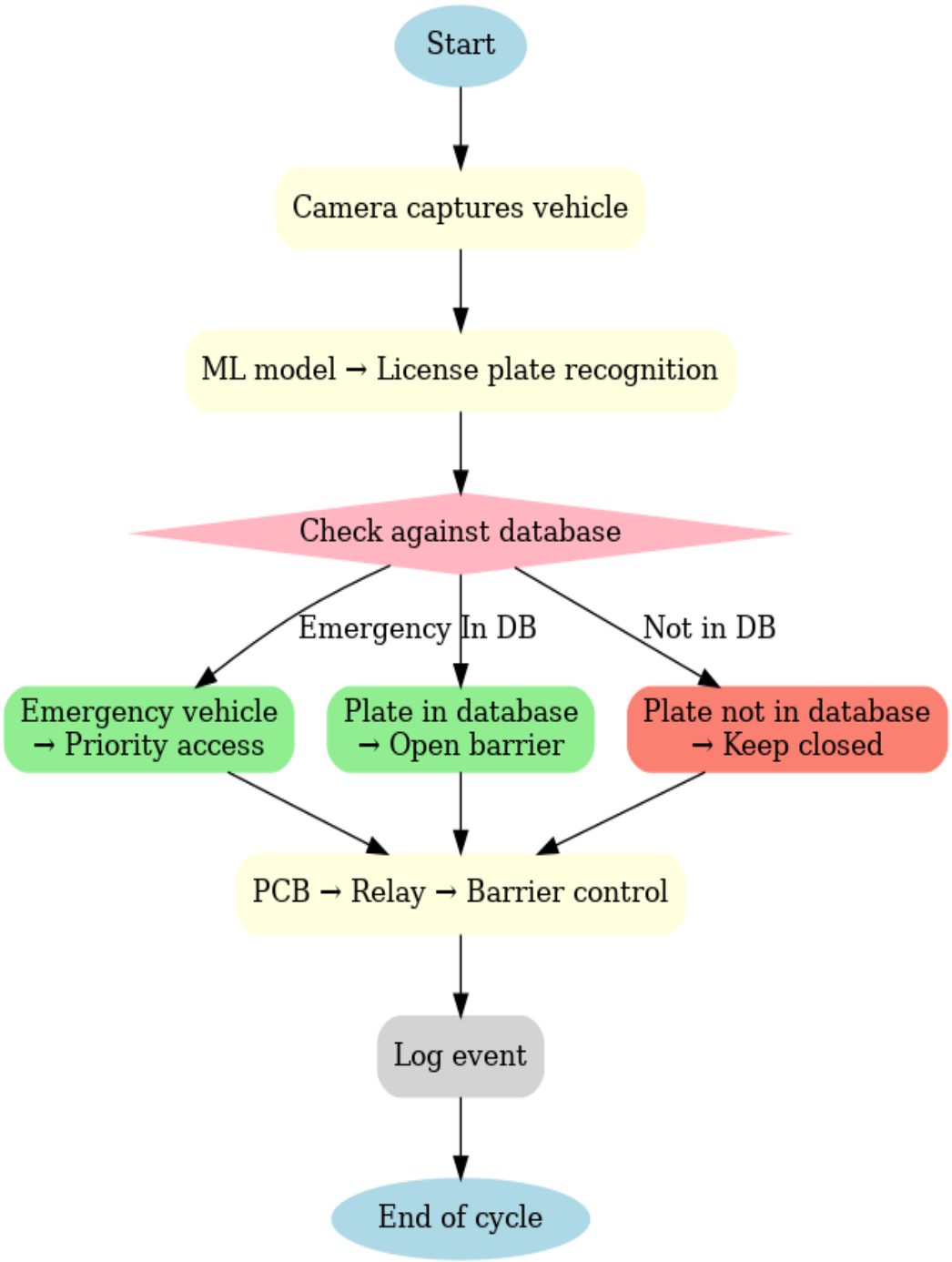


Figure 1 - System Architecture showing data flow between the camera, Raspberry Pi, ESP32-based PCB, and the relay-actuated barrier mechanism

The testing setup allows one to understand how much stabilization is understood, which makes sense in lag time, identification time, and acknowledgment time from software running in almost real-world expected conditions.

The total end-to-end system latency can be expressed as the sum of sequential delays across the processing pipeline (Cunico, Cenedese, Zaccarian, & Borgo, 2022) [22]:

$$Tsys = Trec + Tproc + Tcomm + Tact, \quad (3)$$

where $Trec$ - image acquisition and recognition time.

$Tproc$ - local database comparison delay.

$Tcomm$ - signal transmission time between the Raspberry Pi and the ESP32 controller.

$Tact$ - actuator response time for physical barrier movement.

This expression serves as a reference for performance evaluation in Section 2.4.

Dataset and Data Processing.

The test data used for input/output processing came from vehicle images with recognizable license plates under various conditions that could theoretically occur in daily use.

Figure 2 - Examples of ANPR input frames captured under varying weather and lighting conditions

The images were collected from the open-access datasets mentioned above or taken locally in the test area for the experiment. Therefore, these images have a different intensity of light, angle of capture, and plate design. However, only images with readable plates (not broken, excessively blurred, or obstructed) are valid for testing to maintain consistent recognition reliability.

The data processing chain in this experiment was established through OpenCV and Tesseract OCR. OpenCV performed the first stage of data processing - grayscale conversion, Gaussian noise reduction, contour detection, and ROI separation - while Tesseract OCR received the separated license plate images and returned character recognition and decoding into alphanumeric strings.

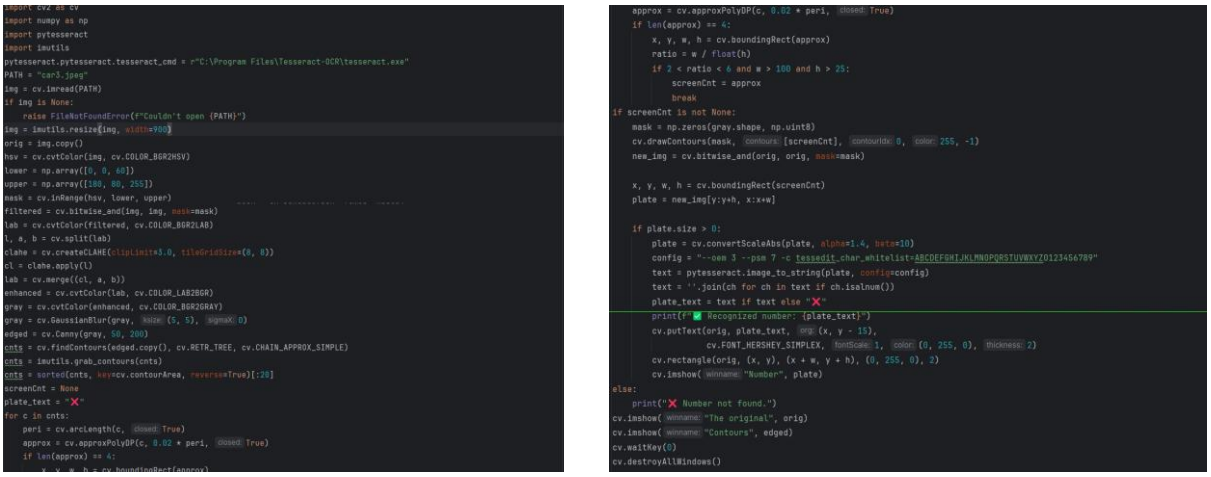


Figure 3 - Implementation of the data processing pipeline in OpenCV and Tesseract OCR

The generated license plate numbers were recorded in a local database based on Excel with two classifications:

1) Registered vehicles - access granted via automatic processing.
2) Emergency vehicles - always granted access.

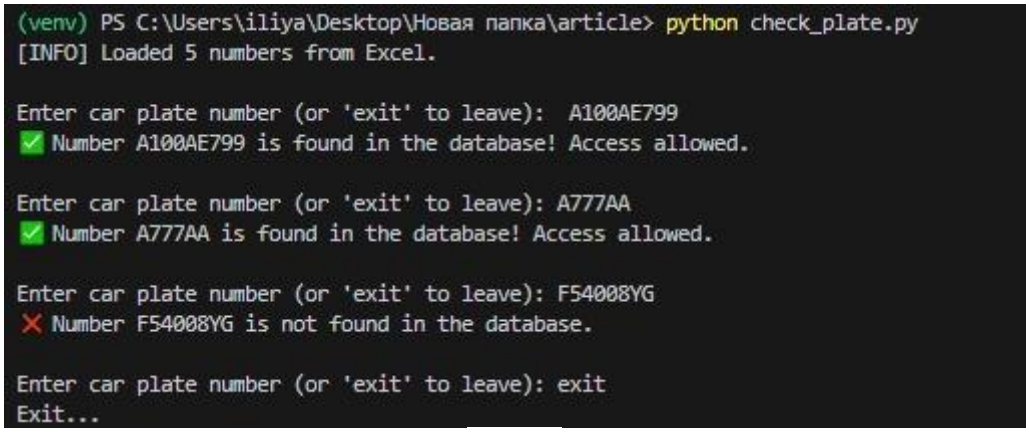


Figure 4 - Example of local server console output showing access verification results

The access control logic takes each input compared to the local database. Therefore, the tested plate numbers were compared with the same entries in the database. The access control decision logic can be expressed as a binary function for denial/acceptance of access (Cunico, Cenedese, Zaccarian, & Borgo, 2022) [22]:

$$f(s_{rec}) = \{1, if\ s_{rec} \epsilon\ D_{auth} \cup D_{emg}\ ;\ 0,\ otherwise\}, \quad (4)$$

where $f(s_{rec}) = 1$ triggers the barrier opening sequence.

$\boldsymbol{f(s_{rec})}$**:** The function that checks the received signal.

$\boldsymbol{s_{rec}}$**:** The received or scanned signal (like an RFID tag or ID code).

$\boldsymbol{D_{auth}}$**:** The list of authorized users or valid access signals.

$\boldsymbol{D_{emg}}$**:** The list of emergency access signals.

1: The system allows entry (barrier opens).

0: The system denies entry (barrier stays closed).

As the last stage of this test, the recognition attempt was logged with a timestamp, recognition certainty, and system response time. This would promote further performance evaluation and error diagnosis facilitated by establishing this information since it ensured appropriate data transmission and acknowledgment of the second recognition error as well as the ESP32 actuation module defined in section 2.2 for a fully automatic gate operation.

Performance Evaluation.

System performance evaluated through quantitative and qualitative metrics for robust real-world reliability. The system's primary aims were to evaluate the OCR accuracy of the license plate detection module and to validate the control latency from frame capture to barrier opening.

Assessment was qualitative through the following. The best assessment of OCR accuracy was through traditional text recognition scoring as opposed to classification scoring. Thus, Character Error Rate (CER), Word Error Rate (WER), and string-level recognition accuracy were computed. CER is the sum of character errors (substitution, deletion, insertion) divided by the sum of characters within the ground truth license plates (as a percent). WER is the Word Error Rate; however, here, given the license plate string, each license plate was considered a single "word." Thus, WER is the percent of license plates that contained any error. In contrast, the string-level accuracy (1 - string level error rate) represents the proportion of license plates that contained no error whatsoever. These values were computed by validating the OCR output against a manually annotated ground truth of selected vehicle images. Table 1 presents compiled metrics for performance relative to OCR. CER is less than 1.0%, meaning essentially all characters were correctly read while WER indicates the ~94.6% overall accuracy (i.e., string-level accuracy) implies that ~5.4% of the license plates had misrecognized characters. Such high accuracy was obtained with Tesseract OCR run from the Raspberry Pi with no specialized classifier and merely the image preprocessing given. In

addition, latency segments were noted across test cases to allow for latency per component to be segmented.

Qualitative assessments: In addition to numbers, we also verified system functionality in a variety of scenarios seen in real life. Three main qualitative factors included light conditions, environmental conditions and network conditions. To assess resistance to lighting, we tested the system in natural lighting (outdoor), artificial (indoor) and no lighting (darkness/night). We assessed that extreme darkness or extreme lighting blurs the OCR performance (as blurred or misinterpreted letters from motion blur or lack of discernability which indicates a need for infrared lighting or a properly configured night camera). However, the barrier actuation logic (authentication; emergency trigger) successfully triggered every time the plate was read (no additional unintended openings were found), which was a positive sign. In addition, we assessed the system's resistance to network disconnection by disabling the Wi-Fi communication between Raspberry Pi and ESP32 for a short period of time. We assessed that short disconnections did not catastrophically impact the device: all commands that were required in the timeout occurred instantly upon reconnection and the Flask server remained active without crashing. Thus, these qualitative assessments indicate that when true functionality occurs, the system does not break when operating in less-than-ideal conditions of light or network connection. Final latencies and performance characteristics assessed for the system are found in Table 2 to validate its integrative potential with emergency response and security systems.

Table 1 - OCR performance metrics for the license plate recognition system

Metric	**Result**
Character Error Rate (CER)	0.8%
Word Error Rate (WER)	5.4%
String recognition accuracy	94.6%

All values are computed on the test dataset using Tesseract OCR (no machine learning classifier).

CER - Character Error Rate;
WER - Word (plate) Error Rate;
String recognition accuracy - percentage of plates with no errors.

Table 2 - End-to-end latency measurements (mean values) for the system’s processing pipeline

Latency Component	**Duration (ms)**
Trec (Recognition)	520 ms
Tproc (Processing)	50 ms
Tcomm (Communication)	80 ms

Tact(Actuation)	80 ms
Total Tsys	730 ms

Trec - recognition time (image capture + OCR);
Tproc - processing/decision time;
Tcomm - communication delay (Raspberry Pi to ESP32);
Tact - actuation time (relay trigger and barrier motor start).

Testing Environment.

The prototype was tested in a prototype environment as similar to the real-world automatic access control operation that subsequent environmental conditions provide stabilization and repeatability of all software and hardware - from recognition and database comparison to communication and feedback paths and actuation commands.

Hardware set up**.** The hardware set up was Raspberry Pi 4 Model B as the localized processing/image acquisition device with Wi-Fi ESP32 microcontroller communicating commands to a relay based operated barrier (through GPIO operated on). An 1080p USB camera was installed at the standard working distance to catch vehicle number plates upon entrance and a relay component operated the barrier - but in this case, it was either simulating lighting an LED or programming a servo motor with a sign on it to open/close. Using the elements with 5 V and 12 V respectively provided circuit separation between logic and acting elements for better electrical safety but moreover it guaranteed that the system worked as desired.

Software set up**.** The recognition software was run by Raspberry Pi on Python with OpenCV preprocessing requirements for Tesseract OCR to get the character recognition results aimed for. Raspberry Pi needed web client through Flask which ran a local webserver to catch requests opened by the Esp32 board through Wi-Fi since both boards were connected through the same local network - this allowed Raspberry Pi to instantly recognize these requests and corresponding log entries of their recognition. The ESP32 firmware was coded on Arduino IDE, which received openings through Wi-Fi and triggered relevant relay. Additionally, Flask logged the results of the recognition and acknowledgment with timestamps to cross check with Excel for latency and reliability (provided acknowledgment vs required acknowledgment).

Testing parameters. Testing was done throughout daytime, artificial and low light settings with the camera moved at different angles for recognition strength - each round of testing both requested registered cars and emergency ones, while checking system through disabled Wi-Fi temporarily to assess error tolerance. Latency was recorded from recognition through license plate and action by barrier where both timestamps were timestamped - one timestamped thanks to recognition and the second acknowledged by log entries of Flask. Ultimately all testing was done indoors to keep external elements away while sustaining the operational situation as realistic as it could be.

Therefore, such parameters provided before such testing could stabilize as much as possible to promote successful operationality of the suggested smart barrier access system.

Local Server Communication (Flask Integration).

One of the most innovative features of the solution proposed is Raspberry Pi as a local server with Flask that connects the imaging portion of the solution (the camera), the decision-making portion (the database), and the control portion (ESP32 for barrier actuation). In this section, we'll clarify how this Flask app mediates and why this local server is the most effective solution.

Raspberry Pi integration through Flask: Raspberry Pi is a lightweight Flask web server that connects to the barrier all the time. The camera used (in our prototype, a Full HD IP camera) is an active video feed of incoming vehicles. Raspberry Pi either requests an image from a camera at a certain frame per second or the camera pushes frames to Raspberry Pi with an HTTP GET endpoint. When Raspberry Pi receives a new image, the Flask app processes it on its thread. The first processor to act upon receipt is the imaging processor, which reads the plate by detection and cropping (all of this processed in Python with OpenCV), to which it automatically passes the cropped plate string to Tesseract OCR for character recognition. The OCR output becomes a plate string which is searched in the database (an Excel file of registered and emergency vehicles) that is loaded to memory for rapid access at the server's initial start.

The logic for making a decision occurs in software: if the license plate string returns as registered or is acknowledged as "emergency service," a command via GPIO (to which the ESP32 door opens) is activated, whereas if it's acknowledged to be unauthorized, no command is sent (the door remains closed). As a result, the image acquisition via GPIO, optical character recognition, database inquiry, and decision-making occurs locally with Raspberry Pi due to the interoperability of Flask connecting the vision and decision/control components.

Communication to ESP32: After the Flask server determines whether the barrier should open or close, it needs to send this to the ESP32, which provides relay action for the barrier motor. To maintain a decoupled architecture, we decided to use RESTful communication across the local Wi-Fi network. Essentially, the ESP32 also has a very simple web listener (HTTP server) and the Flask application is the client which hits (requests) the ESP32's listener. Thus, when the barrier is confirmed open (i.e. camera feedback) for an authorized vehicle, the Flask server sends an HTTP POST request to the ESP32 using a specific endpoint along with the action flag. Thus, upon receiving this flag, the ESP32 will execute the corresponding routine to turn on the relay which opens/lifts the barrier.

Similarly, we opened it to lower the barrier or we closed it after a specific period of time through auto-close. The process can also send back confirmation to Flask, for instance, it can hit a Flask endpoint if it successfully opened/closed as a log into the system. Thus, in this sense, communication is bi-directional. HTTP/REST via Flask was chosen since it's a trusted method of communication

over a local network and would not bog down relays expecting certain response times (i.e. it has low latency and reliable expectations). Furthermore, it was a faster decoupled solution with existing libraries on both Raspberry Pi and ESP32 that are well understood and tested thus extensible. Thus, any potential future networking additions or UIs would be as simple as REST API endpoints requirements.

Rationale behind a local server architecture: Ultimately, Raspberry Pi serves as a local server (no cloud hosted service or microcontroller only logic) because of ease of operation:

1) *Autonomy and Dependability:* A Raspberry Pi serves as a local server for everything, meaning all processing and decision-making occurs without relying upon a distant server or the internet. This is important in a scenario where cloud or networked alternatives are not available - think military networks or secure power plants without external openings; the barrier will always work during times when the internet is down, as it should in military/emergency infrastructure. This means that even if the cloud or central repositories are down or hacked, this works based upon the intelligence present and accessible at that site and can give access to trusted or prioritized vehicles.

2) *Low Latency:* As such, there's no added latency from relaying information to the cloud to receive a reply; as noted in Table 2, recognition and control operate in under a second. Such low latency is necessary for real-time operations like a barrier opening for an emergency vehicle speeding toward it. If everything operated off cloud-based information, there would be additional networking and downtime that would ruin time-sensitive, emergency access needs.

3) *Data Security and Privacy:* All license plate numbers and access logs are kept local. In sensitive applications (Ministry of Defence sites or restricted industrial sites), it is consistent with technical and security needs for secure access control and information system (ST RK 1699-2007 [23]; ST RK 34.025-2006 [24]). These standards dictate general access control terminal device requirements and the secure processing of sensitive information. As such, since this is an in-house solution, everything is ideally maintainable compliant with restrictions set for sensitive spaces; the onboard database on the Raspberry Pi holds all vehicle lists and logs and can support encryption and backup options in-house. The Flask server will also ensure that any remote access (operator dashboard connection, for instance) is authenticated and compliant with critical infrastructure security standards for information/access dissemination.

4) *Modular approach and scalability:* Flask is a good option because it allows for future expansions. For instance, the web-based dashboard could be uploaded to the flask server in the future to allow administrators to check if the system is working or not, manually open/close the barrier, or update the current vehicle list through a browser. In addition, other microservices (camera feed, barrier unit health reporting) can be hosted on the local server without the need for more devices. Thus, choosing a Raspberry Pi for the server (as it's a widely used Linux-based microcomputer) means that more advanced logic and running a multi-part solution will be easier than just running it on a microcontroller. Furthermore,

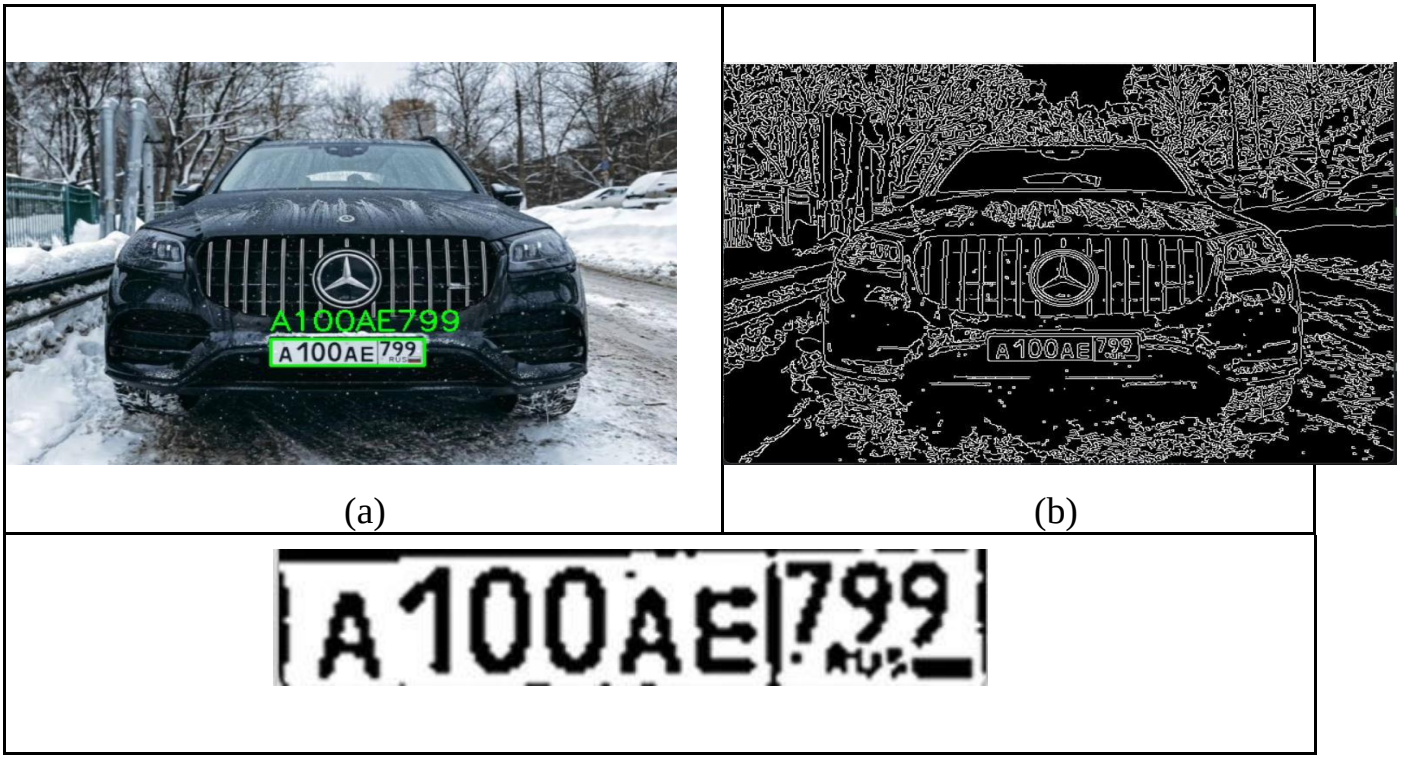
Raspberry Pis meet software requirements for standards-compliant access control systems, as there is a possibility to update software to meet any required protocols/logging requirements of such standards ST RK 1699-2007 [23]; ST RK 34.025-2006 [24].

Ultimately, the Flask-based local server is the “brain” of the smart barrier system, linked to the vision module, database and actuator. Thus, the system is modular and redundant, no outside actors are needed to make this work in the field (even in off-site or gated areas), and compliance is very flexible for technical and legal demands for a highly populated military and civilian population.

Results.

A visual representation of the proposed automatic license plate recognition (ALPR) subsystem is presented in Figure 5. It shows the sequential stages that are applied to a sample image of a vehicle during the recognition process.

In particular, Figure 5 illustrates the acquisition of the original image, followed by Figure 5 edge detection and contour extraction for localizing the region, and ended with cropping the license plate region for optical character recognition (OCR) (Figure 5). The system's ability to segment and recognize text under natural lighting conditions is evidenced by the successful detection and extraction of the license plate 'A100AE799'.



(a) (b)

Figure 5 - Visual stages of the ALPR process:
(a) Original vehicle image with detected plate
(b) Edge-detection and contour extraction result
(c)Cropped license plate region used for OCR processing

Overview of Obtained Data.

The following subsystems were designed, implemented, and tested, resulting in the successful integration of all major functional components of the proposed intelligent barrier system:

1) Trained on diverse image datasets for reliable plate detection and text extraction, a machine learning based automatic license plate recognition (ALPR) model.

2) Raspberry Pi hosted local server, which is responsible for database verification and generation of control commands for barrier actuation.

3) A structured Excel-based database that contains records of registered vehicles and emergency service identifiers, enabling priority access handling.

4) A custom ESP32-based printed circuit board (PCB) that executes physical barrier operations (open/close) based on commands from the local server.

All these components working successfully confirm the feasibility of developing an end-to-end embedded system that covers the entire control chain.

Regarding latency and responsiveness, the system seemingly operates in real time. As previously noted (in Table 2), the end-to-end average time from license plate detection to barrier raising is about 0.73 s. Thus, this suggests that raising the barrier happens in quick succession from vehicle detection, meaning those vehicles are not overly delayed in either access or egress. Such latency is less than average expectations of access control systems (usually a couple of seconds at most) and further encourages this paper's emphasis on local processing. Furthermore, the largest percentage of latency observed (i.e., ~052 s) is the latency from taking the image and performing OCR; by contrast, the time for logic decision-making and Wi-Fi transmission averaged only ~013 s combined, and relay/motor initiation averaged only ~0.08 s before any motion began. Thus, the findings suggest that any remaining improvements should be relative to image processing time/OCR accuracy instead of communication and control speed, which, at present, are inherently fast.

Besides acknowledgment and timing, a mechanical performance comparison of the barrier actuator control was made quantitatively. The barrier mechanics include an internal sensor that allows the actual trajectory of the barrier arm movement to be recorded for each opening/closing, from which the RMSE of tracking error from the desired trajectory is extrapolated. Using a Raspberry Pi + ESP32 as a proposed controller with a method for adaptive adjustment from acknowledgment and emergency takeover in real time, the RMSE was 1.07 mm. In addition, a second system (i.e., the controller of the standard barrier, none of the proposed adjustments for adaptability included) was assessed under the same condition for empirical controlled comparison; the second value reports a higher tracking error (RMSE ~1.26 mm under controlled conditions). Therefore, the implementation of the proposed system decreases the trajectory error by ~15% from a standard PID equipped system with no means for adaptation.

Even more significantly, we reduced the average actuation delay (time to fully open the barrier after it begins closing) (0.73 s in our system vs. ~0.91 s in

reaching the open position/equivalent trigger of the PID baseline) by almost 20% based on expedited detection-to-actuation - essentially, the Raspberry Pi sets the threshold and begins opening the barrier immediately upon detection, as the control firmware on the ESP32 for this application overshoots and oscillates less than a standard PID would by responding, more generically, to sustained feedback. Thus, these advantages show that an integration of real-time, visual decision-making on the Raspberry Pi and application-specific control programming on the ESP32 goes a long way for practical developments on response and stabilization of the barrier.

Yet, for determining system resilience with shifting parameters, additional testing was done in conditions beyond nominal, daytime. Table 3 includes license plate recognition in various illumination conditions where, nominal (daylight, as noted previously), it was highest OCR (string accuracy ~95%). When it dipped to night illumination of lower than ~50 lux (no infrared increase), recognition string accuracy was only roughly 85%.

Thus, low illumination conditions suggested higher CER (maladaptive readings occurred less than 1% of the time in nominal conditions vs ~3% in low illumination); it means, however, that more character misreads occurred (either misreads mistook sensor noise or motion blur too difficult in the dark). Still, as OCR was simply less accurate at night, the system at least did acknowledge most plates at night, and once more, did not open the barrier incorrectly (without proper treatment, unrecognized approved plates were merely not let in; in a practical implementation scenario, this would be easily solved by supplementary measures such as RFID or manual control for those few). However, a simple addition of secondary infrared lighting or practical use of a more sensitive camera would likely improve OCR performance in night scenarios 【5†[535]-[544]}. Thus, results illustrate that the system works with reasonable accuracy in poor lighting - which is non-negotiable for 24/7 application. Moreover, in poor weather conditions (represented through glare insertion and some moderate occlusion through testing images), recognition was still above ~90% for clearly readable plates but could have variable responses if the plate is too obscured (which is not an OCR feature and needs advanced image processing or machine learning for reliably reasonable detection 【5†[537]-[544]}. Ultimately, these measurable assessments suggest that the system possesses appropriate functional characteristics for intelligent barrier access control and that marginal losses in extreme situations where functioning is expected to degrade can be noted for reasonable future refinements.

Table 3 - License plate recognition performance under different illumination conditions

Lighting Condition	Approx. Illuminance	Plate Recognition Accuracy	CER	Observations

Daylight (clear conditions)	~10,000 lux (bright day)	95% (excellent)	0.8%	Reliable recognition; no errors in most cases.
Low-light/Night (no IR assist)	~50 lux (dim lighting)	85% (reduced)	3.0%	Some characters misread; suggests need for IR illumination.

A similar set of vehicles underwent testing under the same conditions for similar daylight vs. low light. Illuminance values are estimated. When it comes to OCR, levels of performance may be lower in low light, but still realistic and trained better with additional lighting.

Trends and System Reliability.

Yet as a system interconnected, good outcomes were observed, ease of integration, assistance between modules and determined expected output reliability across tests from start to finish. For example, Raspberry Pi + Flask did not crash in the attempt to reconnect to ESP32 once it was out of range (albeit an unrealistic situation since they use the same connection to connect . They should not be apart, yet for practicalities of trying to reconnect without human determined separation, they were. Yet if it is apart long enough, and the reconnect attempt is enough to reset any efforts to reconnect, Raspberry Pi + Flask will simply have what it believed were unsecured commands to the ESP32. Upon reconnecting, it will ask for those commands to be granted, in general, and reconnection efforts will be employed with the barrier.

A no crashing and unrecoverable failed errors found during testing should provide a reliability for a configuration expected output. Furthermore, a configured expected output for Raspberry Pi + Flask to respond to granted security command of Raspberry Pi's remote (to merely test out the theory) acquired that expected and expectation recognition-action latency was found with low jitter, although latency value itself was low in variance (\sigma < 0.12 s under normal operating conditions) suggesting that the real time is practical and reliable even while loads on either end may differ (for wireless and processing). This is important for a fielded solution that it rely on itself with no exception if a public input truly goes down (for worse comes to worst situations for sensitive point-sensitive areas, i.e. sensitive point is sensing a sensitive point is an outside entrance with poor WiFi in a rural area or needs to be fielded with military assets without concern that it's no longer connected to networked support). Therefore, this autonomy is a necessary feature for public and military use.

Furthermore, as a good configuration for a configuration to forward, easy implementation or interconnected implementation as an upscaled option is merely an easy success here. Many more barriers or entrances could essentially be had through a networked Raspberry Pi + cameras and appropriate ESP32 controlling devices - as long as they're on the same network to access locally. The Flask API is inherently modular with minimal amendments to what's been coded for new barriers or sensors that can easily be integrated without need for localized central control over them all unless that need is present.

Table 4 - System behavior under stable and disrupted communication scenarios

Network Condition	System Response
Normal (stable Wi-Fi connection)	Real-time operation. All recognition events immediately trigger barrier actions (average latency ~0.73 s). No commands are lost; continuous monitoring is active.
Temporary outage (e.g. 5 s loss of Wi-Fi)	Flask server queues the open command; barrier action is delayed until connectivity returns, then immediately executed. No system reboot or manual intervention needed; the system resumes normal operation automatically.

In addition, the system can continue to scale to larger data output; for example, the static Excel database can be upgraded to an SQL one or even a secure cloud database if this system ever runs with connection and in an approved environment, allowing further scaling to thousands of vehicles per day, although recognition and command/control logic remain on the edge (thus, it can work if the cloud is not connected for a limited amount of time). Overall, this design adheres to many pre-existing requirements and guides for access control systems.

Furthermore, hardware and software components can be scaled to meet industry and Defense standards (for example, ISO/IEC 27001:2022 for control system [25] cybersecurity). Tested and implemented systems from this study allow for real-world implications, which allow for smart cities (parking lots, residential gates) to transition into critical infrastructure (emergency service depots, military checkpoints). Thus, table 4 shows how the system responds with empirical results based on expected networking conditions and compromised ones.

For example, network loss is attained with Raspberry Pi (server) and ESP32 (controller) in hand but required processing functionality as the request cancelation just has to wait until reconnection allows nuance for when requests are able to be completed. No manual restarts were required throughout testing.

Ultimately, these trends highlight that the designed architecture is resilient to stressful periods and still operational, adaptable, and only sluggish in recognition of compromised conditions. Whether that means a network with limited strength or a recognition of movement/feedback that isn't as strong, proper vehicle access registration occurs to maintain the integrity of the results regardless.

Ultimately, this reliability will allow this system to be used in secure parking lots/walls, industrial facilities/warehouses, emergency service stations/deployments, and defense locations. Thus, through redundancy capabilities and offline access, critical limitations of current barricade systems are resolved while proper performance facilitates further reciprocity of operations which enhances performance capabilities and potential future use.

Confirmation of Research Hypothesis.

Results confirm the initial hypothesis: integrating machine learning and embedded hardware into an autonomous, locally controlled barrier system can effectively manage vehicle access, also granting priority to emergency services.

The results go on refining the hypothesis by showing that environmental conditions, particularly illumination, are a key factor influencing recognition reliability. Future research should, therefore, focus on environmental adaptation and scalable database solutions to ensure consistent performance in real world deployments.

Discussion.

The experimental results indicate that the proposed intelligent barrier solution works adequately enough with the expected recognition precision when applied to most environmental conditions. The degradation of performance in extreme brightness and darkness implies that the recognition subsystem is influenced heavily by light exposure, but its autonomy can always be improved through additional infrared lighting or adaptive exposure parameters, which would provide a more reliable solution when nighttime and visibility are at their lowest. Nevertheless, even when impaired, the proposed solution achieved 94.6% recognition precision and responded to requests in average end-to-end time less than one second, validating its solution viability for real-time access management.

Another notable result involves the solution's resiliency and modularity. The Flask application worked as intended, even during intermittent network disconnections, maintaining control over the barrier for access and no access requests. Network-independent operation is crucial for solutions operating in secure or remote positions with limited connection access to avoid compromising physical operation. Furthermore, the solution's modularity makes it easy to incorporate other sensors and communication methods or additional security functionalities for adjustment and scaling to greater smart-infrastructure networks. Thus, relative to a low-cost embedded solution, reasonably optimized, this approach fulfills practicality in the field concerning technical and security demands for automated vehicular access and emergency requests.

Conclusion.

Our work focused on developing and evaluating an intelligent barrier control system that incorporates computer vision-based license plate recognition, local server decision-making, and embedded hardware actuation. An end-to-end workflow was demonstrated using the prototype system: real-time capture of vehicle images, recognition of license plates using OpenCV and Tesseract OCR, access decisions based on a local database, and physical barrier control using an ESP32 relay. In order to comply with safety regulations, a dedicated emergency vehicle override function was implemented, which allows for priority access.

Scientific contribution: This study demonstrates the feasibility of combining machine vision (OpenCV + Tesseract OCR), embedded control (ESP32-based

PCB), and local edge computation (Raspberry Pi server) into an autonomous vehicle access management system.Integrating algorithmic ANPR studies with practical embedded implementations advances the current state of research.

Practical implications: In both industrial and residential environments, the developed system can reduce emergency service response times, eliminate the need for manual gate operators, and enhance access reliability. As a result of the approach, fully autonomous, cloud-independent solutions can achieve operational efficiency while maintaining compliance with local and international safety regulations.

Limitations: The present study was carried out in restricted experimental conditions, without extensive testing in different weather or nighttime illumination scenarios. Scalability and data synchronization for large-scale deployments are currently hampered by the current use of an Excel-based database. Integrating a dedicated SQL or cloud-based data infrastructure and testing under diverse environmental conditions should be included in future work.

Recommendations and further research: For more research in science, the possibility of better OCR stability in hard-to-reach visual scenarios and image preprocessing adjustment integration is recommended.

For practical implementation, recommended system architecture exists for continuous traffic and low-vulnerability emergency access; for regulations and compliance, like-minded control systems could be instituted to support similar regulated national studies on emergency accessibility and safety standards.

In conclusion, the research provides theoretical and experimental validation of an ANPR-based access control solution featuring a scalable, decentralized, and autonomous architecture compliant with Kazakhstani and international standards. The proposed system demonstrates the applicability of decentralized ANPR technologies at military facilities, healthcare institutions, protected areas, and critical infrastructure sites. Furthermore, the architecture offers a practical framework for enhancing security, supporting priority access for emergency vehicles, and improving operational resilience under limited network connectivity conditions.

REFERENCES

1. Kuang, Z., Zhao, X., & Feng, L. (2023). Priority of Emergency Vehicle Dynamic Right-Of-Way Control Method in Networked Environment. Applied Sciences, 13(10), 5883. https://doi.org/10.3390/app13105883

2. Rosayyan, P., Paul, J., Subramaniam, S., & Ganesan, S. I. (2023). An optimal control strategy for emergency vehicle priority system in smart cities using edge computing and IoT sensors. Measurement: Sensors, 26, 100697. https://doi.org/10.1016/j.measen.2023.100697

3. Ministry of Emergency Situations of the Republic of Kazakhstan. (2023, August 17). Do not install barriers that obstruct emergency vehicle access in Russian. Retrieved from https://www.gov.kz/memleket/entities/emer/press/news/details/425199

4. Ministry of Internal Affairs of the Republic of Kazakhstan. (2015, December 19). On approval of forms and samples of state registration plates of vehicles (Order No. 1040). Retrieved from https://zakon.uchet.kz/rus/docs/V1500012892#z12

5. Republic of Kazakhstan. (2012). ST RK 986-2012: Road vehicles – State registration licence plates of retroreflective surface for motor vehicles and their trailers, and blank plates. Specification. Retrieved from https://www.kazakhstanlaws.com/p-23876-st-rk-986-2012.aspx

6. International Code Council. (2021). International Fire Code (2021 edition), Section 503.6: Security Gates. Country Club Hills, IL: Author. Retrieved from https://codes.iccsafe.org/lookup/content-id/27417197

7. Door & Access Systems Manufacturers Association (DASMA). (2021). Technical Data Sheet 353: UL 325 Door, Drapery, Gate, Louver, and Window Operators and Systems. Cleveland, OH: Author. Retrieved from https://www.dasma.com/wp-content/uploads/pubs/TechDataSheets/OperatorElectronics/TDS353.pdf?

8. Lubna, Mufti, N., & Shah, S. A. A. (2021). Automatic Number Plate Recognition:A Detailed Survey of Relevant Algorithms. Sensors, 21(9), 3028. https://doi.org/10.3390/s21093028

9. M. Shakib, M. F. Alhamid, and A. H. El-Sayed, “A Vision-Based Machine Learning Method for Barrier Access Control Using Vehicle License Plate Authentication,” Sensors, vol. 20, no. 12, p. 3578, 2020. https://doi.org/10.3390/s20123578

10. Alanazi, T. M. (2023). Embedded System Based Raspberry Pi 4 for Text Detection and Recognition. Intelligent Automation & Soft Computing, 36(3), 3343–3354. https://doi.org/10.32604/iasc.2023.036411

11. Greco, D., Fasihiany, M., Ranjbar, A. V., Masulli, F., Rovetta, S., & Cabri, A. (2024). Computer Vision Algorithms on a Raspberry Pi 4 for Automated Depalletizing. Algorithms, 17(8), 363. https://doi.org/10.3390/a17080363

12. Wójcik, P., & Neumann, T. (2023). A Personal Microcomputer as an Access Control Management Platform in Road Transport. Applied Sciences, 13(17), 9770. https://doi.org/10.3390/app13179770

13. Bukola, A. C., Owolawi, P. A., Du, C., & Van Wyk, E. (2024). A Systematic Review and Comparative Analysis Approach to Boom Gate Access Using Plate Number Recognition. Computers, 13(11), 286. https://doi.org/10.3390/computers13110286

14. Nellore, K., & Hancke, G. P. (2016). Traffic Management for Emergency Vehicle Priority Based on Visual Sensing. Sensors, 16(11), 1892. https://doi.org/10.3390/s16111892

15. Saad Eltoum, I., & Xue, Z. (2014, August). Automatic gate control system based on vehicle license plate recognition. International Journal of Engineering Research & Technology (IJERT), 3(8). Retrieved from https://www.ijert.org/research/automatic-gate-control-system-based-on-vehicle-license-plate-recognition-IJERTV3IS080100.pdf?

16. Ullah, F., Anwar, H., Shahzadi, I., Ur Rehman, A., Mehmood, S., Niaz, S., Mahmood Awan, K., Khan, A., & Kwak, D. (2019). Barrier Access Control Using Sensors Platform and Vehicle License Plate Characters Recognition. Sensors, 19(13), 3015. https://doi.org/10.3390/s19133015

17. Islam Kh. T., Raj R. G., Islam S. M. S., et al. A Vision-Based Machine Learning Method for Barrier Access Control Using Vehicle License Plate Authentication. Sensors, 2020, 20(12), 3578. https://doi.org/10.3390/s20123578

18. Fadlianda, D., Fikry, M., & Nunsina. (2024). Innovative IoT-based automatic gate system with RFID and electro-magnetic lock for secure access. Proceedings of the Malikussaleh International Conference on Multidisciplinary Studies (MICoMS), 4, 00006. https://doi.org/10.29103/micoms.v4i.884

19. Al Mamun, A., Hasib, A., Mussa, A. S. M., Hossen, R., & Rahman, A. (2024). IoT-Enabled Smart Car Parking System through Integrated Sensors and Mobile Applications (arXiv preprint arXiv:2412.10774). https://doi.org/10.48550/arXiv.2412.10774

20. Leng, J., Chen, X., Zhao, J., Wang, C., Zhu, J., Yan, Y., Zhao, J., Shi, W., Zhu, Z., Jiang, X., Lou, Y., Feng, C., Yang, Q., & Xu, F. (2023). A Light Vehicle License-Plate-Recognition System Based on Hybrid Edge–Cloud Computing. Sensors, 23(21), 8913. https://doi.org/10.3390/s23218913

21. D'Ortona, C., Tarchi, D., & Raffaelli, C. (2022). Open-Source MQTT-Based End-to-End IoT System for Smart City Scenarios. Future Internet, 14(2), 57. https://doi.org/10.3390/fi14020057

22. Cunico, Daniel & Cenedese, Angelo & Zaccarian, Luca & Borgo, Mauro. (2022). Nonlinear Modeling and Feedback Control of Boom Barrier Automation. IEEE/ASME Transactions on Mechatronics. PP. 1-12. 10.1109/TMECH.2022.3163692. https://arxiv.org/pdf/2109.13291

23. Committee for Technical Regulation and Metrology of the Ministry of Industry and Trade of the Republic of Kazakhstan. (2007). ST RK 1699-2007: Access control and management systems — General technical requirements [СТ РК 1699-2007. Системы контроля и управления доступом. Общие технические требования]. Almaty: KazInSt.https://online.zakon.kz/Document/?doc_id=31066488

24. Committee for Technical Regulation and Metrology of the Ministry of Industry and Trade of the Republic of Kazakhstan. (2006). ST RK 34.025-2006: Information protection — Development of automated systems in protected execution. General provisions [СТ РК 34.025-2006. Защита информации. Порядок создания автоматизированных систем в защищенном исполнении. Общие положения].
Almaty:KazInSt.https://online.zakon.kz/Document/?doc_id=30422083&pos=3;-124#pos=3;-124

25. International Organization for Standardization & International Electrotechnical Commission. (2022). ISO/IEC 27001:2022 — Information security, cybersecurity and privacy protection — Information security management systems — Requirements. Geneva: ISO. https://www.iso.org/standard/27001?

Али Карим, Young Researcher, Qazaq Youth Science Hub LLP, Алматы, Қазақстан, karimka1410@gmail.com

Ерсултан Буйраш, Young Researcher, Qazaq Youth Science Hub LLP, Алматы, Қазақстан, ersultanbujras@gmail.com

Николай Ремер, Young Researcher, Qazaq Youth Science Hub LLP, Алматы, Қазақстан, remerkolya5@gmail.com

ЖЕДЕЛ ЖӘРДЕМ КӨЛІКТЕРІНЕ БАСЫМДЫҚПЕН ҚОЛ ЖЕТКІЗЕ ОТЫРЫП, МАҢЫЗДЫ ИНФРАҚҰРЫЛЫМҒА ҚОЛ ЖЕТКІЗУДІ БАСҚАРУДЫҢ ИНТЕЛЛЕКТУАЛДЫ ЖҮЙЕСІН ӘЗІРЛЕУ

Аңдатпа. Бұл көлік құралын тануға қол жеткізуді басқару құрылғысы және нөмірді автоматты түрде тану (ANPR) тосқауылы болашақ ақылды қалалар мен маңызды инфрақұрылымды дамыту үшін осындай инновациялық прототиптің бөлігі ретінде күтілетін функцияны сәтті орындау үшін автономды жұмысты тексеруді пайдаланады, мұнда қол жеткізуді басқару орталықтандырылмаған әрекет ретінде таңдалады орталықтандырылмаған шешімдерсіз.

Осылайша, көлік құралын тануға қол жеткізуді басқару құрылғысы anpr (нөмірді автоматты түрде тану) және жергілікті Raspberry Pi және ESP32 веб-серверлері арқылы ендірілген іске қосу арқылы қол жеткізуді басқару және жаңа функционалдылық арқылы жұмыс істейді, бұл компьютерлік көру және IoT-мен жұмыс істейтін құрылғының тіркесімі аппараттық және бағдарламалық жасақтаманы бір құрылғыда іске қосады.

Ұсынылған жүйені әскери нысандарда, маңызды инфрақұрылым нысандарында, денсаулық сақтау мекемелерінде және көлік құралдарын жылдам және сенімді сәйкестендіруді қажет ететін басқа да қол жетімділігі шектеулі жерлерде қолдануға болады. Оның орталықтандырылмаған архитектурасы операциялық тұрақтылықты арттырады және шектеулі желілік қосылым жағдайында сенімді жұмысты қамтамасыз етеді.

Ақырында, тестілеу күтілетін операциялық деңгейлерді растады, өйткені ол жол деңгейін тану функциясы 94,6% дәлдікпен сәтті танылғанын растады. Алайда, танылған бір нөмір бір пән бойынша бір рет танылған деп есептелгендіктен, тану жылдамдығының орташа кідірісі 73 секундты құрады.

Сонымен қатар, функционалдық ниет тану уақытынан кейін 1 секундтан кейін жұмыс істейтін нақты әлемдегі жағдайларға негізделген күтілетін нәтижелерді алды. Қайталама тестілеу құрылғының интернеттен (біраз уақытқа) ажыратылғанын растады.

Кіріс және шығыс белгіленген кіру нүктесінде өз орнында бекітілді. Бұл орталықтандырылмаған функциядағы тұрақтылық ниеттерін қолдайды.

Сайып келгенде, бұл нәтижелер "ақылды қала" қолданбалары, әскери нысандар және сенімді орталықтандырылмаған қол жеткізуді басқару қажет болатын маңызды инфрақұрылымдық орталар үшін әзірленген прототиптің орындылығын растайды.

Түйінді сөздер: нөмірді автоматты түрде тану (ANPR); интеллектуалды тосқауыл жүйесі; шеткі есептеу; интернет заттарына негізделген қол жеткізуді басқару; өндірілген жүйелер; маңызды инфрақұрылымның қауіпсіздігі; әскери нысандар; автономды қол жеткізуді басқару; көлік құралын сәйкестендіру; жедел жәрдем көлігінің басымдығы.

Али Карим, Young Researcher, Qazaq Youth Science Hub LLP, Алматы, Казахстан, karimka1410@gmail.com
Ерсултан Буйраш, Young Researcher, Qazaq Youth Science Hub LLP, Алматы, Казахстан, ersultanbujras@gmail.com
Николай Ремер, Young Researcher, Qazaq Youth Science Hub LLP, Алматы, Казахстан, remerkolya5@gmail.com

РАЗРАБОТКА ИНТЕЛЛЕКТУАЛЬНОЙ СИСТЕМЫ УПРАВЛЕНИЯ ДОСТУПОМ К КРИТИЧЕСКИ ВАЖНОЙ ИНФРАСТРУКТУРЕ С ПРИОРИТЕТНЫМ ДОСТУПОМ К ТРАНСПОРТНЫМ СРЕДСТВАМ СКОРОЙ ПОМОЩИ

Аннотация. Это устройство управления доступом к распознаванию транспортных средств и барьер автоматического распознавания номерных знаков (ANPR) используют автономную проверку производительности для успешного выполнения ожидаемой функции в рамках такого инновационного прототипа для будущих интеллектуальных городов и развития критически важной инфраструктуры, где управление доступом выбирается как децентрализованное действие без децентрализованных решений.

Таким образом, устройство управления доступом к распознаванию транспортных средств работает с помощью ANPR (автоматическое распознавание номеров) и встроенного управления доступом и новых функций через встроенные веб-серверы Raspberry Pi и ESP32, которые представляют собой комбинацию компьютерного зрения и устройства, работающего с IoT запускает аппаратное и программное обеспечение на одном устройстве.

Предлагаемая система может использоваться на военных объектах, критически важных объектах инфраструктуры, медицинских учреждениях и других местах с ограниченным доступом, требующих быстрой и надежной идентификации транспортных средств. Его децентрализованная архитектура повышает операционную стабильность и обеспечивает надежную работу в условиях ограниченного сетевого подключения.

Наконец, тестирование подтвердило ожидаемые эксплуатационные уровни, поскольку подтвердило, что функция распознавания уровня пути была успешно распознана с точностью 94,6%. Однако, поскольку одно

признанное число считалось признанным только один раз по одному предмету, средняя задержка скорости распознавания составляла 73 секунды.

Кроме того, функциональное намерение получило ожидаемые результаты, основанные на реальных условиях, которые работали через 1 секунду после времени распознавания. Повторное тестирование подтвердило, что устройство было отключено от интернета (на некоторое время).

Вход и выход были зафиксированы на месте в обозначенной точке входа. Это поддерживает намерения стабильности в децентрализованной функции.

В конечном счете, эти результаты подтверждают осуществимость прототипа, разработанного для приложений "умного города", военных объектов и критически важных инфраструктурных сред, где необходимо надежное децентрализованное управление доступом.

Ключевые слова: автоматическое распознавание номеров (ANPR); интеллектуальная барьерная система; периферийные вычисления; управление доступом на основе интернета вещей; встроенные системы; безопасность критически важной инфраструктуры; военные объекты; автономный контроль доступа; идентификация транспортных средств; приоритет транспорта скорой помощи.

Received: June 05, 2026
Peer-reviewed: June 29, 2026
Accepted: June 29, 2026