Intelligent Entry Control

ERIK DOVGAN, MATJAŽ GAMS
Jožef Stefan institute
Jamova cesta 39
SLOVENIA
erik.dovgan@ijs.si

Abstract: Entry control is an important issue for security and optimization reasons. Input sensors based on biometrics and intelligent methods that learn from experience are used to recognize terrorists or simply to detect an unusual behavior of regular stuff. We have designed and developed an intelligent entry system consisting of four independent modules: expert system, micro learning, macro learning and camera. In the experimental set-up, there are four input sensors: a door sensor, an identity card reader, a fingerprint reader and a camera. Each of the four modules produces an explanation that categorizes an event as alarm or normal, and the system proposes the final suggestion with explanation.

Key-Words: Ambient intelligence, access control, event classification

1 Introduction

Intelligent methods help an administrator with alarm messages when a risky situation occurs at entry point [1]. To protect an important object, intelligence provides important improvements [2]. It is important to have a full access control realized with sensors on every entry point of the object and an intelligent system that processes every event and categorizes it as an alarm indicating a terrorist thread or a normal event [3]. It is also important that the system recognizes a strange behavior of the employee when, e.g., stealing or tailgating [4].

Intelligent methods control human behavior on two levels. At micro level, each person has its own micro timing when accessing the building, depending on the person's ability and habits. For example, if a person has its identity card in the wallet it differs as if he or she has the card in the pocket. Some persons do the identification and verification very quickly while others do it slowly. Some people also throw open the door while others open the door just to slip through. This behavior usually does not change in time and is a good identification of a person [5].

At macro level, control is based on the daily routine of a person: the time a person enters the building depends on day and week. Similarly, smokers usually exit the building more often than non-smokers. This behavior also does not change much in time. Based on micro and macro characteristics, the intelligent system can classify an event depending on the user behavior.

2 Entry-control system

2.1 Architecture

We had set-up an experimental environment that consists of a door with a sensor that detects when the door is opened and when it is closed, an identity card reader, a fingerprint reader and a camera as shown in Fig. 1.

Fig. 1: One of security-threat scenarios. On the right side there are an identity card reader and a fingerprint reader.

Every regular event must consist of successful card identification and fingerprint verification [6]. If identification and verification are successful, the door unlocks. For every regular event the following four times are registered in the central database:

- time of acceptance of the identity card
- time of acceptance of the fingerprint
- time of door opening
- time of door closing.

The sensors send a message to the system to inform that an event has occurred. All modules of the intelligent system then start to process the event data. Each module produces classification of the event [7]. The final classification is based on all the four modules as shown in Fig. 2. Each module can categorize an event as:

- OK: the access is regular
- Warning: it is unclear whether the access is regular or not
- Alarm: the access is irregular.

All the modules perform the classification simultaneously. The sequence of results coming from the modules is not defined. Faster modules produce results before slower modules. The results are shown in the sequence they are produced. Besides the classification, every module also produces an explanation of its decision that is also shown on the screen.

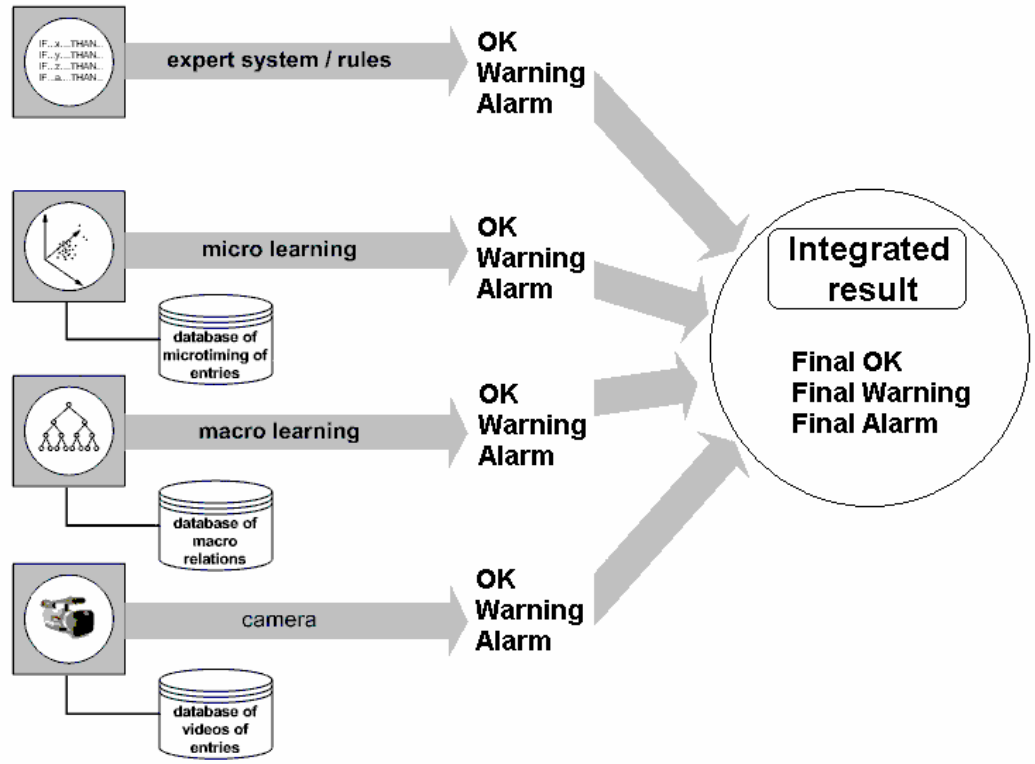

Fig. 2: The architecture of the system.

2.2 All modules

The intelligent system consists of four modules:

- expert-defined rules
- micro learning
- macro learning
- camera.

Expert rules define what a formally legal entry is. Micro learning learns user's past behavior at the micro lever and classifies an event according to it [8]. Local outlier factor algorithm is used for event classification.

Macro learning learns user's past behavior at the macro lever and classifies based on it. A tree is used for classification for event classification.

Camera learns user's past behavior from the video and classifies an event regarding body and arm movement [9].

3 Modules

3.1 Access and database

In our prototype, four sensors were used. These are: door sensor, identity card reader, fingerprint reader and camera. The number of such features can vary depending on desired complexity of the system. The input signals from identity card reader, fingerprint reader and a door sensor are collected by DOX – a multi-channel access controller shown in Fig. 3, developed by the company Špica International. A single DOX controller can collect signals from only one access point. If there are more access points, multiple DOX controllers have to be combined.

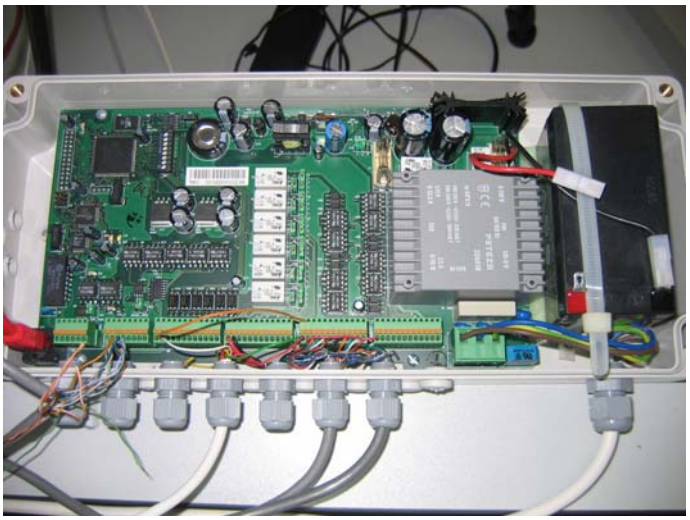

Fig. 3: DOX – a multi-channel access controller.

The intelligent system uses Time&Space database. Every sensor event is collected by a Time&Space program Event Collector. It processes the event and writes a row in the database that has all the information about the event. If the identity card event occurs, event collector writes date and time when it happened, type of event that is either entry or exit and user identification. If the fingerprint event occurs, event collector writes date and time when it happened and the accuracy of fingerprint verification. If the door event occurs, event collector writes date and time when it happened. The last kind of event occurs twice in a regular sequence, when the door opens and when it closes. After every regular sequence, entry or exit, there are four new rows in the database:

- identity card row
- fingerprint row
- door opened row
- door closed row.

During every sequence of events the Event collector produces two signals. The first signal passes through TCP/IP communication channel. This signal is received by the intelligent system that was previously registered in the Event collector for receiving a message whenever a regular sequence of events occurs. The system listens to the TCP/IP communication channel as a client while the Event collector is a server on port 4444. The event collector sends a message that a regular sequence of events occurs. Our intelligent system then retrieves all other data about the event from the centralized database. This kind of message is called a Clocking event.

The second signal that the Event collector produces passes through the message queue to the Video collector. This is another Time&Space program that controls the camera. When the Video collector receives the message, it reads pre-buffered video sequence from the camera and starts to collect images from the camera for few seconds. Then it joins the pre-buffered video sequence with collected images to a video which is saved to the disk.

Event collector passes a signal though TCP/IP to our system also when a non regular sequence of events occurs. This happens if the user opens the door without fingerprint verification or when he or she opens the door without card identification and fingerprint verification. In the last case the door closing is not needed. The signal is fired when the door opens, e.g., when there is an explosion. This kind of message is called Alarm event.

3.2 Rules

Expert rules do not learn from the behavior of a person. The user defines them from a set of predefined generic rules. He or she chooses a generic rule and defines its parameters. Generic rules are:

a) a warning/alarm is fired if user produces an event between time1 and time2 on a day
b) a warning/alarm is fired if user produces more than a number of events in a time
c) a warning/alarm is fired if user does not exit in time
d) a warning/alarm is fired if all users leave the building between time1 and time2
e) a warning/alarm is fired if time between event1 and event2 is greater than time (event is identity card or fingerprint or door opening or door closing)
f) a warning/alarm is fired if user did not exit before time
g) a warning/alarm is fired if user produces doubled entry or exit.

Underlined words are parameters of generic rules that have to be defined before a generic rule becomes a functional rule as shown in Fig. 4. Every rule is tested every time that is called by the main program thread except the rule f. This rule is called on timer every ten seconds.

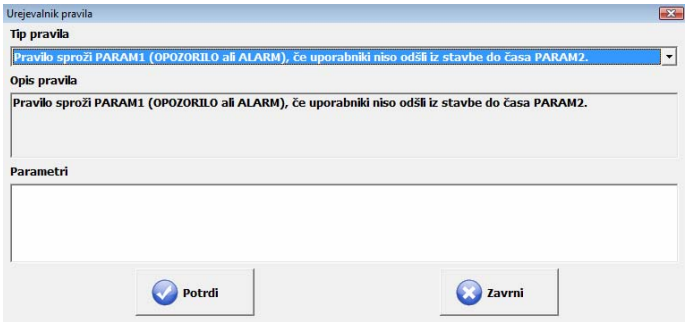

Fig. 4: Form for adding a new rule.

Every rule produces a conclusion and an explanation. Explanations are showed on screen in an instant when

they are produced. The final result combines results from all the rules. If one of the rules produces an alarm as its result, then the final result is an alarm. If there are no alarms, then if one of the rules produces warning as its result, then the final result is a warning. If there are no alarms and no warnings, then the final result is ok.

3.3 Coordination

When a new event occurs, a message is send through TCP/IP communication channel. The system reads the message and starts to process this new event. If the sequence of records in the database that describes an event is not a regular sequence, than the event is an irregular event. The sequence is: identity card, fingerprint, door opened, door closed. The message from TCP/IP can arrive when the sequence is not finished yet. If the message is a Clocking event, the system waits till all the events of the sequence occurs and afterwards it starts to process the event. If the message is an Alarm event, the system does not wait for events of the sequence because the sequence is not regular by definition. The processing of the event starts immediately. It also does not wait for door closing if a special rule is enabled. With this rule we can define the maximum time that the door can be opened. If the door is opened more then the predefined time, the system stops to wait for door closing and starts to process the event.

The intelligent system reads event rows from the database. Then it calls four threads, one thread for each intelligent module. The main thread stops and waits till all the four threads exit.

Any of these modules can be disabled due to whatever reason. This means that when it is called, it produces a null result. When producing the final result, a null result is threatened as an alarm.

The expert-system module is an inside module of the system. It is called by the expert-system thread which waits till the module returns the result.

The micro learning module is a stand-alone module. The program produces an input file for this module that contains all the data regarding the micro learning. Bounds for warning and alarm and the number of neighbors are defined in the program. These arguments are passed to the micro module when it is called. The intelligent program waits for the end of the micro processing. The micro module produces a file with probability of normal event and an explanation. This information is read by the intelligent program and displayed on the screen.

The macro learning is a stand-alone module consisting of tree modules:

- macro tree learning
- micro tree learning
- micro macro LOF learning.

The program produces tree input files, one for each module that contains all the data regarding the macro learning [10]. Bounds for warning and alarm and the number of neighbors for LOF and the final result are defined in the program. There is also an importance of each module which is used when combining results from the modules to the final result. These arguments are passed to the macro module when it is called. The intelligent program waits till the macro processing ends. The macro module produces a file with results of each module, its explanation and the final result. This information is read by the intelligent program and displayed on the screen. The result is produced by comparing the number in the macro result and the bounds for warning and alarm.

The module that processes videos from camera is a stand-alone module. The main program communicates with camera module through the TCP/IP communication channel. The camera module acts as a server that listens to the port 8888. The main program connects to the camera module as client and when new event occurs, a request message is send to the camera module. The main program thread for the camera stops and waits till the camera module sends a result to it.

When all four threads exit, the main program combines the four results of these modules into the final result and displays it on the screen as shown in Fig. 8. The final result is ok or warning or alarm. We can define how many modules must produce a type of result that the final result will be the same type. We can choose between 1 and 4. If we choose for example 3, then at least 3 modules must answer ok that the final result is ok. If this condition is not satisfied, then at least 3 modules must answer ok or warning that the final result is warning. If even this condition is not satisfied, then the final result is alarm.

Only one event can be examined at once. If another event occurs it produces its own thread that has to wait till all the other events are examined.

4 An example event

The user enters the building at the approximate time as usual. Firstly he must pass the card identification check point. He does that with his personal identification card. The card reader's sensor senses and reads the card information and sends a message to the Time&Space controller. The controller queries the database to check if the card number exists in the user's table. If no match is found, the access is denied and the entry sequence is discarded. Otherwise the controller saves the user identification number into its memory and waits for fingerprint verification. The event information is saved to the database. This information is date of event, time of

event, user identification number and access point identification number. If the user skips the card identification, an alarm message is send to the Time&Space controller. It recognizes the strange event and does not unlock the door.

Secondly he must pass the fingerprint verification check point. He must lay his index finger on the fingerprint reader. Fingerprint reader's sensor scans the finger and sends a message to the Time&Space controller. The controller reads the user identification number from its memory and retrieves user's fingerprint information from the database. Then the similarity of fingerprint, read at the fingerprint sensor, and fingerprint, stored in the database, is calculated. If the fingerprints match enough, the access is allowed. The event information is saved to the database. This information is date of event, time of event and similarity of fingerprints. The controller stores to its memory the information, that a regular event occurred and unlocks the door. Otherwise the access is denied and the entry sequence is discarded. If the user skips the fingerprint verification, an alarm message is send to the Time&Space controller. It recognizes the strange event and does not unlock the door.

Thirdly he must open the door. The door can be opened only if there were a successful identification and verification. Otherwise the door is locked. If the door is unlocked and the user opens it, the door sensor senses the event and sends a message to the Time&Space controller. The door opening can also occur if there is a bomb attack that destroys the door so the door opens. The controller must recognize if the door opening is part of a regular access or if it is an unusual event. It does that by reading its memory where the last access information is stored. If a regular sequence of events occurred in a short past time period, then the door opening is part of a legal event sequence. The controller sends the event information to the database, where it is stored. This information is date of event and time of event. It also sends a request for recording to the camera program. The camera records constantly and saves the video to its memory. Because of camera's limited memory space only a few seconds of recorded video can be stored in the memory. The video storing procedure is called video buffering. When a new picture is captured by the camera, the oldest picture in the memory is replaced by this new picture. Camera program sends a request for the video to the camera. Camera sends the pre-buffered video to the camera program and recording the video for a few seconds. At the end of the recording this video is also send to the camera program. Camera continuous buffering the video and listening for the new requests. Camera program receives two videos: the pre-buffered video and the new video. It saves these videos to the disk.

If an irregular sequence of events occurs, for example, if there is no identification or verification, then the controller sends an alarm message to the Event collector which passes it to the intelligent system. No information is stored in the database and camera program does not query videos from camera.

At the end the user must close the door. This event is sensed by the door sensor. When the door closing event occurs, the door sensor sends a message to the Time&Space controller. The controller recognizes the signal and sends event information to the database, where it is stored. This information is date of event and time of event. It also sends a regular event message to the Event collector which passes it to the intelligent system. The closing door event is the last event in a regular event sequence. If there is no closing door event in a defined time period, starting from the door opening, the event sequence is interrupted. The controller sends an error message to the Event collector and the error event information to the database, where it is stored.

The intelligent program listens for event messages. The messages are sent by the Event collector. When a message arrives, the program reads it, creates a new main thread and passes the message to it. Only one main thread can exists at one time so only one access can be processed at one time. All other possible accesses must wait till the current access handling ends.

The main thread reads the type of the access from the message. It is an alarm access or a regular access.

A regular access is produced when all four access events occur in the valid sequence. The main thread reads the access information from the database and tests if the access is a regular access. The access can be irregular even if the message says that a regular access occurs. The sensors are not 100% accurate. A sensor can sense an event and send a message with a delay so the time of the event, stored in the database, can be corrupted. For example, even if there is a regular event, the storing of card event can occur later than the storing of fingerprint event. The same thing can happen with the door sensor. Corrupted events cannot be processed because of that, so the rules are not tested, micro and macro learning are not called and camera module is not queried. The main thread displays the wrong sequence event explanation on the screen. Then it ends and the program can start to process another event.

If the regular event is not corrupted, the intelligent program can start to process it. It is processed by four modules: rules, micro learning, macro learning and camera module. Each of them can be turned off. The main thread stops and waits till all four modules produce a result.

If the rules are turned on, the main thread creates a rule thread and passes it the identification number of the user, who produced the access. This thread tests every

rule that is defined by the administrator. Most of the rules are a single user rules. They are valid only for a specific user. The thread tests if the rule is a single user rule and if it is, it tests if the rule is defined for the user who produced the access. If this condition is satisfied, then the rule is tested. Otherwise the test of the rule is skipped. Each rule tests a specific thing, for example, a rule can test, if the number of accesses, produced in past X minutes, is greater than Y. The rule thread retrieves all the needed data for the rule testing from the database [11]. If the rule is not a single user rule, then it is valid for all users and it is always tested. That kind of rule may also need additional information from the database so the rule thread must query the database. Each rule produces a result that is ok or warning or alarm. If one of tested rules produces an alarm, then the final result is alarm. If there is no alarm and one of tested rules produces a warning, then the final result is warning. If there is no alarm and warning, then the final result is ok. When all the rules have been tested, then the final result is send to the main thread and the rule thread ends. In the described example all rules classified the event as ok so the final result is ok.

If the rules are turned off, then no rule testing occurs. The final result of the rules is ok.

If the micro learning is turned on, then the main thread creates a micro learning thread and passes it the identification number of the user, who produced the access. The micro learning thread reads the user past events and the user current event from the database. For each of these events the thread retrieves times of card, fingerprint, open door and close door events. Then the differences between card time and fingerprint time, between fingerprint time and open door time and between open door time and close door time are calculated for each event. Differences of user past events define the learning events, differences of user current event define the test event. Learning events are stored into the learn events file. Current event is stored into the test event file. The micro learning module is a stand-alone module. It acts as a server that processes requests received through TCP/IP communication channel. The micro learning thread sends a TCP/IP request message to the micro learning module. It sends the paths of learn events file, test event file and the number of neighbors (N) – a parameter of micro module. Then it stops and waits for the response that is also send through TCP/IP communication channel. Micro learning module receives a message and retrieves file paths and number of neighbors from it. Then it reads learn events from the learn events file and test event from the test event file. Local outlier factor algorithm is used to classify the test event regarding to learn events as shown in Fig. 5.

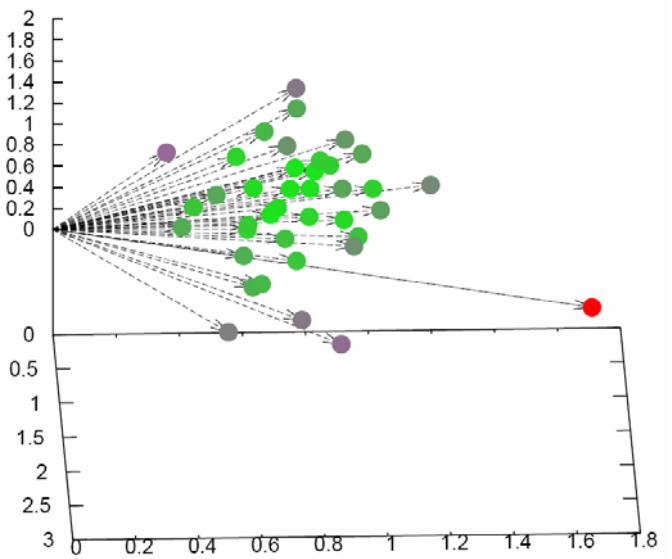

Fig. 5: Every event is presented as a point in the three-dimensional space. Local outlier factor calculates distances between each pair of events. Events, that are closer together are classified as OK, those far away are classified as alarm [12].

It finds N nearest learn events to the test event, calculates distances between pairs of chosen learn events and distances between each chosen learn event and the test event [13]. If the mean distance between pairs of chosen events is similar to the mean distance between each learn event and the test event, then the event is classified as OK. If the mean distance between each learn event and the test event is much greater, then the event is classified as alarm. The event, described in the example, is far away from other events as shown in the Fig. 5. It is presented as a red point, so it is classified as alarm [14]. The result is send to the micro learning thread through TCP/IP communication channel. The micro module continuous to listen for the new messages. The micro learning thread reads the result from TCP/IP communication channel, sends it to the main thread and ends.

If the micro learning is turned off, then no message exchange with micro module occurs. The final result of the micro module is ok.

If the macro learning is turned on, then the main thread creates a macro learning thread and passes it the identification number of the user, who produced the access. The macro learning thread reads the information about the user past events and about the user current event from the database. For each of these events the thread retrieves times of card, fingerprint, open door and close door event.

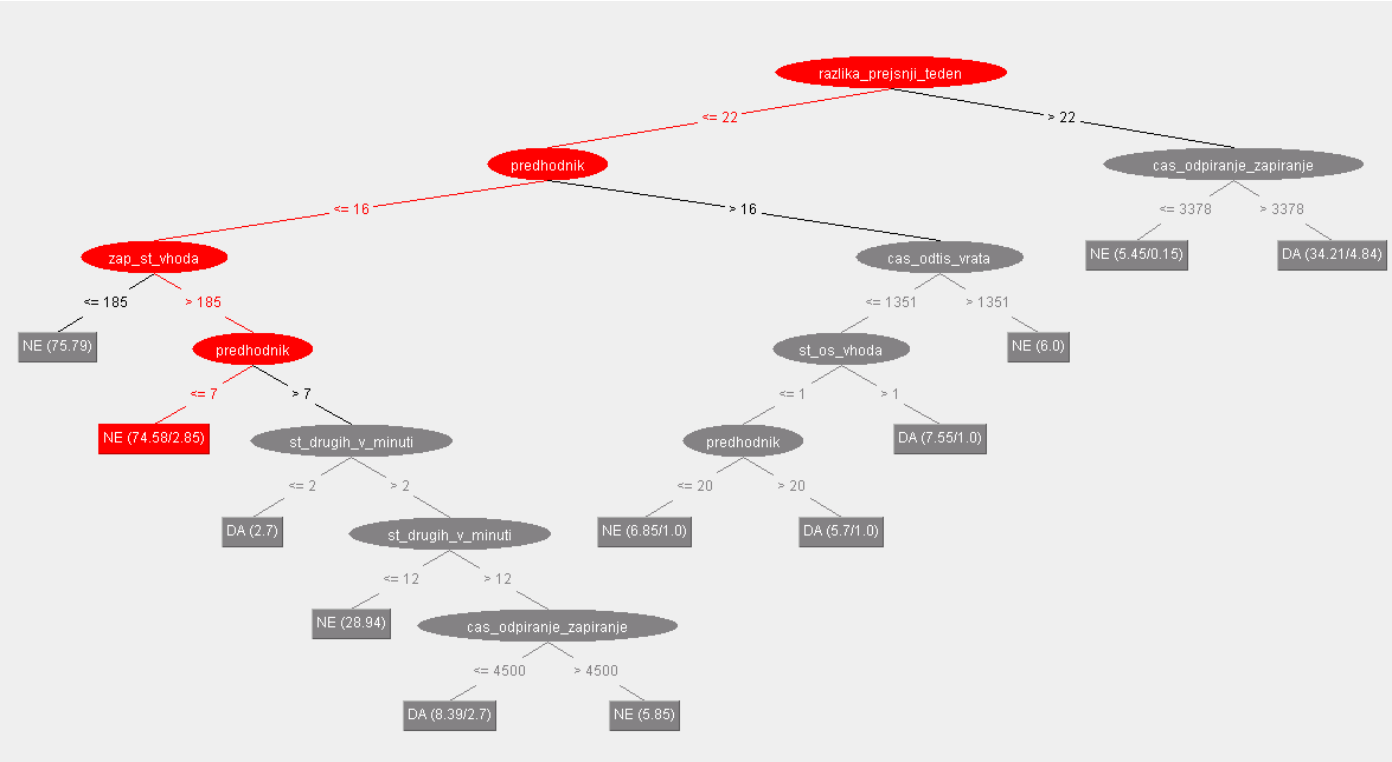

Fig. 6: A tree is build according to the past user's events. Each new event is classified using this tree. Tree attributes are day of week, time, predecessor, and so on.

The differences between card time and fingerprint time, between fingerprint time and open door time and between open door time and close door time are calculated for each event. The thread calculates day of week and time in the day from the open door time. For each event is also retrieved the user that produced the event before the current event. Each event is presented with six values. Values of the user past events defines learning events, values of the user current event defines a test event. Learning events are stored to three learn events files. Current event is stored to three test event files. The macro learning module is a stand-alone module. Three files of each type are produces because the module is made of tree independent sub-modules. The macro learning module acts as a server that processes requests received through TCP/IP communication channel. The macro learning thread sends a TCP/IP request message to the macro learning module. It sends paths for the learn events files, paths for the test event files, the number of neighbors and the weights for each of tree sub-modules. Then it stops and waits for response that is also send through TCP/IP communication channel. The macro learning module receives a message and retrieves all the data from it. Then it reads learn events from the learn events files and test event from the test event files for each of the three sub-modules. It calls each of the three sub-modules and passes them all necessary data. The first module uses local outlier factor algorithm, which is also used by the micro module. Events are represented with all six attributes. Test event is classified as ok or warning or alarm. Each of the other two modules uses classification tree for the event classification as shown in Fig. 6 [15]. The first tree is build considering only day of week, time in the day and predecessor of learn events. The second tree is build considering all six values of learn events. Test event is classified by each tree. Each tree returns the percentage of similarity between the test event and each learn event [16]. The event, described in the example, is classified as alarm as shown in the Fig. 6. The final result is calculated by adding weighted tree percentage of each tree to the weighted result of the first sub-module. This result is ok or warning or alarm, depending on result bounds. The result of the event, described in the example, is alarm. The result is send to the macro learning thread through the TCP/IP communication channel. The macro module continuous to listen for new messages. The macro learning thread reads the result from the TCP/IP channel, sends it to the main thread and ends.

If the macro learning is turned off, then no message exchange with macro module occurs. The final result of the macro module is ok.

If the camera is turned on, then the main thread creates a camera thread. The camera thread sends a TCP/IP message to the camera module and stops until it receives a response from camera module. The camera module is a stand-alone module which acts as a server

that processes requests received through the TCP/IP communication channel. When a message arrives, the camera module reads the last saved video from the disk. It also retrieves the identification number of the last event user from the database. It reads past videos of that user and learns user's usual movements from them as shown in the Fig. 7 [17].

Fig. 7: The camera module recognizes parts of the video where some movement occurs and calculates the intensity of the movement. A new event video is classified based on similarity of other videos [18].

Then it reads the last video and tests if the learn movements are similar to the movements on current video [19]. Depending on degree of similarity the module produces ok or warning or alarm as the final result [20]. In the described example the similarity between the current event and the past events was small so the event was classified as alarm [21]. The result is send to the camera thread through the TCP/IP communication channel. The camera module continuous to listen for new messages. The camera thread reads the result from the TCP/IP channel, sends it to main thread and ends.

If the camera is turned off, then no message exchange with the camera module occurs. The final result of the camera module is ok.

When a module produces a result, it is immediately displayed on the screen. The results of the faster modules are displayed before the results of the slower modules. When the main thread received all four results, it calculates the final result which is also displayed on the screen. Each of four modules produces an explanation of its decision which is displayed on the screen together with the module result as shown in Fig. 8.

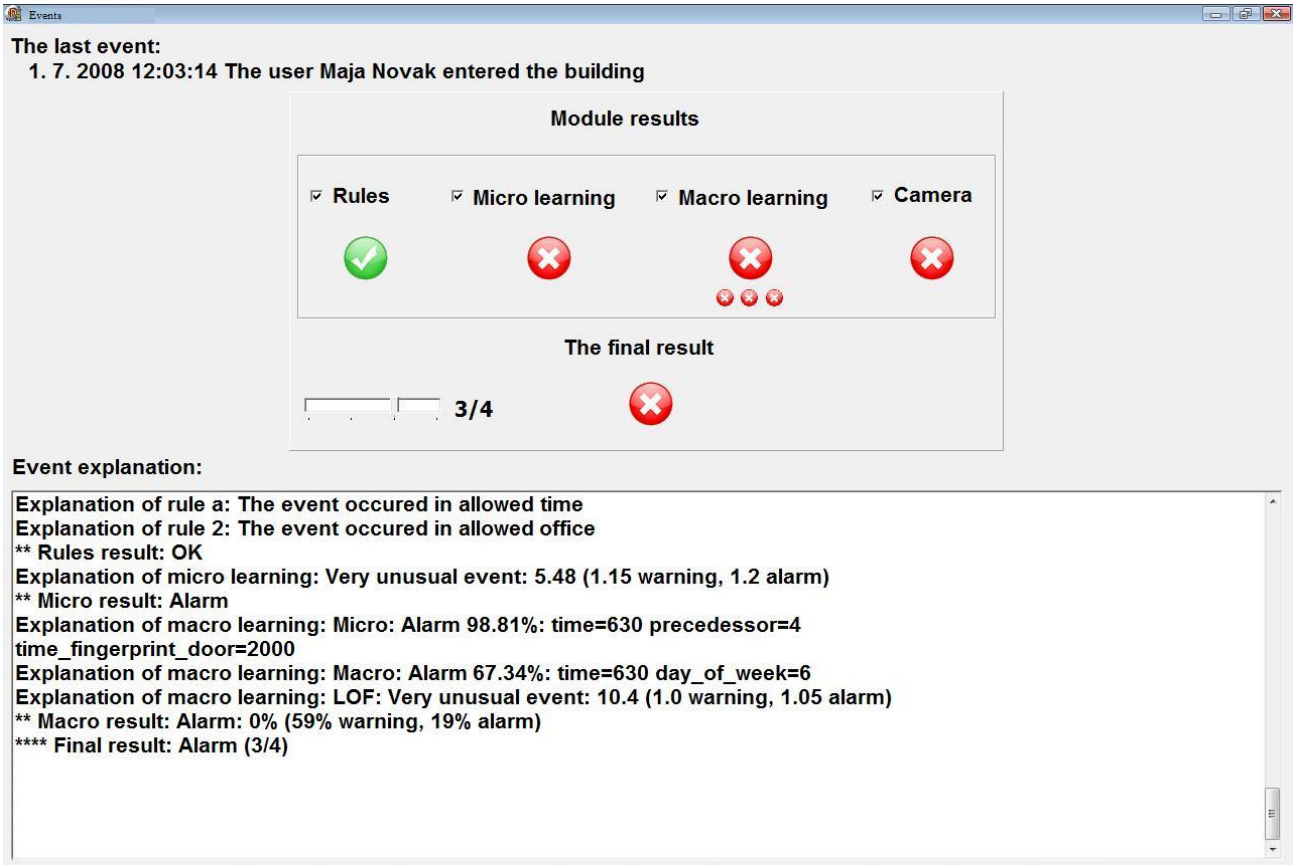

Fig. 8: The classifications of the event, described in the example, are displayed on the screen. The rules result is ok, the micro learning result is alarm, the macro learning result is alarm and the camera result is alarm. The final result is also alarm. At the bottom of the screen the module explanations are displayed.

The event classification ends and the program is ready to receive and classify a new event.

Micro and macro modules produce also a graphical explanation. An example of the micro explanation is shown in Fig. 5. An example of the macro explanation is shown in Fig. 6. These explanations can be displayed on the screen by clicking buttons on the program form.

4 Verification

The test included accesses of five people. Firstly we recorded their entries. Then we simulated the rest of the test. As the simulation of the camera is not simple, we tested only expert-defined rules, micro learning and macro learning.

Each of five tested people made around 40 learn accesses. In the second part of the test each of tested users made another 10 test accesses. We performed the "fake-identity" experiment, in which each person enters into the building with the identification card of another person. In this way, the testing data of one person is also a testing data for other four people.

The results of the experiment are presented in Table 1. Each of the five users entered the building with his or her identification card. Secondly, each of the five users entered the building with each of the four other identification cards.

Results presented in Table 1 show that stand-alone modules have some strong and weak points. The integrated system minimizes weak points of modules and the final result is better than the summary of single modules.

Table 2 and Table 3 show that the integrated system classified OK in 88% of the regular entries and alarm in 69% of the irregular entries.

It should be pointed out that in this experiment all the sensors were simulated as simply by-passed and only the ambient intelligence achieved statistics, e.g. in Tables 2 and 3. In real life, sensors provide basic security and intelligence adds another safety level.

	real	#1			#2			#3			#4			#5			all		
pretender		A	W	Ok	A	W	Ok	A	W	Ok	A	W	Ok	A	W	Ok	A	W	Ok
#1	rules	0	0	10	0	0	10	0	0	10	0	0	10	0	0	10	0	0	50
	micro	0	0	10	1	2	7	10	0	0	4	5	1	10	0	0	25	7	18
	macro	0	3	7	8	2	0	10	0	0	10	0	0	9	1	0	37	6	7
	together	**0**	**3**	**7**	**8**	**2**	**0**	**10**	**0**	**0**	**10**	**0**	**0**	**10**	**0**	**0**	**38**	**5**	**7**
#2	rules	0	0	10	0	0	10	0	0	10	0	0	10	0	0	10	0	0	50
	micro	0	1	9	0	0	10	5	5	0	2	0	8	6	4	0	13	10	27
	macro	10	0	0	0	1	9	1	9	0	0	10	0	10	0	0	21	20	9
	together	**10**	**0**	**0**	**0**	**1**	**9**	**5**	**5**	**0**	**2**	**8**	**0**	**10**	**0**	**0**	**27**	**14**	**9**
#3	rules	0	0	10	0	0	10	0	0	10	0	0	10	0	0	10	0	0	50
	micro	9	1	0	7	1	2	0	0	10	1	0	9	4	4	2	21	6	23
	macro	10	0	0	4	6	0	0	0	10	1	9	0	10	0	0	25	15	10
	together	**10**	**0**	**0**	**7**	**3**	**0**	**0**	**0**	**10**	**1**	**9**	**0**	**10**	**0**	**0**	**28**	**12**	**10**
#4	rules	0	0	10	0	0	10	0	0	10	0	0	10	0	0	10	0	0	50
	micro	4	4	2	0	0	10	0	0	10	0	1	9	0	1	9	4	6	40
	macro	2	7	1	0	4	6	0	0	10	0	0	10	0	0	10	2	11	37
	together	**6**	**3**	**1**	**0**	**4**	**6**	**0**	**0**	**10**	**0**	**1**	**9**	**0**	**1**	**9**	**6**	**9**	**35**
#5	rules	0	0	10	0	0	10	0	0	10	0	0	10	0	0	10	0	0	50
	micro	10	0	0	9	1	0	10	0	0	7	1	2	0	0	10	36	2	12
	macro	10	0	0	9	1	0	10	0	0	10	0	0	0	1	9	39	2	9
	together	**10**	**0**	**0**	**9**	**1**	**0**	**10**	**0**	**0**	**10**	**0**	**0**	**0**	**1**	**9**	**39**	**2**	**9**
all	rules	0	0	50	0	0	50	0	0	50	0	0	50	0	0	50	0	0	250
	micro	23	6	21	17	4	29	25	5	20	14	7	29	20	9	21	99	31	120
	macro	32	10	8	21	14	15	21	9	20	21	19	10	29	2	19	124	54	72
	together	**36**	**6**	**8**	**24**	**11**	**15**	**25**	**5**	**20**	**23**	**18**	**9**	**30**	**2**	**18**	**138**	**42**	**70**

Table 1: Evaluation of the expert-defined rules, micro learning and macro learning on 5 persons.

	A	W	OK
rules	0%	0%	100%
micro	0%	2%	98%
macro	0%	10%	90%
together	**0%**	**12%**	**88%**

Table 2: Statistics for regular accesses.

	A	W	OK
rules	0%	0%	100%
micro	50%	15%	36%
macro	62%	25%	14%
together	**69%**	**18%**	**13%**

Table 3: Statistics for irregular accesses.

5 Discussion

An ambient-intelligence system was designed and tested. The system recognizes terrorists and also strange behavior of regular users. It integrates four independent modules, but in general the number of modules and sensors are arbitrary. The complexity of the system can vary as each of those modules can be turned off. We made some tuning tests and prescribed values that seem to be the most suitable values for our system. Further experimentations have to be done on more real-world data, but even preliminary tests show a significant increase in security.

In summary, the machine learning intelligence that learns from the previous events seems to be an important security mechanism in the war against terrorism.

Acknowledgement

The overall project was funded partly by the Slovenian Ministry of Defense and partly by the Slovenian Research Agency.

References:

[1] Ashbourn, J., *Practical Biometrics: From Aspiration to Implementation*, Springer, 2003.

[2] Turban, E., Aronson, J. E., Liang T.-P., *Decision Support Systems and Intelligent Systems*, Prentice Hall, 2004.

[3] Kolbe, M., Gams, M., *Towards an intelligent biometric system for access control*, in: Proceedings of the 9th International multiconference Information Society - IS 2006, vol. A, pp. 118-122, Jožef Stefan Institute, 2006.

[4] Jain, L. C., Halici, U., Hazashi, I., Lee, S. B., Tsutsui, T., *Intelligent Biometric Techniques in Fingerprint and Face Recognition*, CRC Press, 1999.

[5] Gams, M., Tušar, T., *Intelligent High-Security Access Control*, Informatica, vol. 31, no. 4, pp. 469-477, Slovene Society Informatika, 2007.

[6] Li, C., Yang, Y.-X., Niu, X.-X., *Biometric-based personal identity-authentication system and security analysis*, Journal of China Universities of Posts and Telecommunications, vol. 13, no. 4, pp. 43-47, Elsevier, 2006.

[7] Albus, J. S., Meystel, A. M., *Intelligent Systems: Architecture, Design, Control*, Wiley-Interscience, 2001.

[8] Tušar, T., Gams, M., *Outlier detection in an access control system (in Slovene)*, in Proceedings of the 9th International multiconference Information Society - IS 2006, vol. A, pp. 136-139, Jožef Stefan Institute, 2006.

[9] Kolbe, M., Gams, M., Kovačič, S., Perš, J., *A system for automatic behaviour recognition based on computer vision (in Slovene)*, in: Proceedings of the 8th multiconference Information Society IS 2005, pp. 357-361, Jožef Stefan Institute, 2005.

[10] Zhang, S., Zhang, C., Yang, Q., *Data Preparation for Data Mining*, Applied Artificial Intelligence, vol. 17, no. 5-6, pp. 375-381, Taylor & Francis, 2003.

[11] Breunig, M. M., *Quality Driven Database Mining*, PhD thesis, University of Munich, 2001.

[12] Breunig, M. M., Kriegel, H. P., Sander, J., *LOF: Identifying density-based local outliers,* in: Proceedings of the International Conference on Management of Data SIGMOD '00, pp 93-104, 2000.

[13] Hodge, V. J., Austin, J., *A survey of outlier detection methodologies*, Artificial Intelligence Review, vol. 22, no. 2, pp. 85-126, Springer, 2004.

[14] Kotsiantis, S. B., *Supervised machine learning: A review of classification techniques,* Informatica, vol. 31, no. 3, pp. 249-268, 2007.

[15] Mitchell, T. M., *Machine Learning,* McGraw Hill, 1997.

[16] Quinlan, J. R., *C4.5: Programs for Machine Learning*, Morgan Kaufmann, 1993.

[17] Perš, J., Kristan, M., Perše, M., Kovačič, S., *Motion based human identification using histogram of optical flow,* in: Proceedings of the 12th Computer Vision Winter Workshop, pp. 19-26, University of Technology, Graz, 2007.

[18] Sidenbladh, H., Black, M. J., *Learning the statistics of people in image and video,* International Journal of Computer Vision, vol. 54, no. 1-3, pp.181-207, Kluwer Academic Publishers, 2003.

[19] Wilson, D. L., *Intelligent video systems for perimeter and secured entry access control,* in: Proceeding of the 39th Annual IEEE International Carnahan Conference on Security Technology ICCST, pp. 260-262, 2005.

[20] Yamahara, H., Harada, F., Takada, H., Shimakawa, H., *Dynamic Threshold Determination for Stable Behavior Detection,* WSEAS Transactions on Computers, vol. 7, no. 4, 2008.
[21] Ganorkar, S. R., Ghatol, A. A., *Iris Recognition: An Emerging Biometric Technology*, in Proceeding of the 6th WSEAS International Conference on Signal Processing, Robotics and Automation, Greece, 2007.