Expert Systems with Applications 40 (2013) 3623–3633

Contents lists available at SciVerse ScienceDirect

Expert Systems with Applications

journal homepage: www.elsevier.com/locate/eswa

Recognition of two-dimensional representation of urban environment for autonomous flying agents ☆

Andrzej Bielecki [a,*], Tomasz Buratowski [b], Piotr Śmigielski [c]

[a] AGH University of Science and Technology, Chair of Applied Computer Science, Faculty of Electrotechnics, Automation, Computer Science and Biomedical Engineering, Al. Mickiewicza 30, 30-059 Kraków, Poland
[b] AGH University of Science and Technology, Chair of Robotics and Mechatronics, Faculty of Mechanical Engineering and Robotics, Al. Mickiewicza 30, 30-059 Kraków, Poland
[c] Asseco Poland S.A., Podwale 3, 31-118 Kraków, Poland

ARTICLE INFO

Keywords:
Autonomous flying agents
Vectorization
Object recognition
Syntactic languages

ABSTRACT

In this paper the problem of a vision system implementation for autonomous flying agents is considered in the context of industrial inspection tasks performed by unmanned aerial vehicles. A syntactic algorithm of a two-dimensional object vectorization and recognition is proposed. An algorithm of two-dimensional map recognition has been introduced as well. The algorithms have been tested by using both artificial data and real data – the satellite image. They have turned out to be effective.

© 2012 Elsevier Ltd. All rights reserved.

1. Introduction

Robot vision systems have been studied for over twenty years (Bonin-Font, Ortiz, & Oliver, 2008; Tadeusiewicz, 1992). Such systems have special meaning for autonomous robots, including autonomous flying robots (also referred to as autonomous flying agents or unmanned aerial vehicles-UAV for abbreviation) which are the class of unmanned mobile robots that operate in the air. They are equipped with sensors that are used to collect information about the surrounding environment (Muratet, Doncieux, Briere, & Meyer, 2005; Sinopoli, Micheli, Donato, & Koo, 2001). Beside the need to collect mission-specific data such information enables the robot to find a collision-free path between obstacles (Hrabar, Sukhatme, Corke, Usher, & Roberts, 2005). Identifying their location in space is another typical challenge for the mobile agents (Filliat & Mayer, 2003). Though Global Positioning System (GPS), coupled with other sensors or by itself, has also been extensively used for this purpose, it may not be applicable in environments where there is no satellite visibility, such as under water, in caves, indoors, or on Moon and Mars (Siagan, 2009). In those places vision system should be a viable alternative. Furthermore, complex tasks performed by a robot such as inspection (Metni & Hamel, 2007) or an unknown environment exploring require not only the agent positioning but also the object recognition (Flasiński, 1993; Tadeusiewicz & Flasiński, 1991) and both the scene analysis (Bielecka, Skomorowski, & Bielecki, 2007; Flasiński, 1988; Skomorowski, 1998; Skomorowski, 1999; Skomorowski, 2007) and understanding (see Bielecka et al., 2010; Bielecka et al., 2011; Tadeusiewicz, 2004; Tadeusiewicz, 2006 in the context of medical image understanding). Therefore, systems of pattern recognition and scene analysis based on vision systems have been studied intensively in robotics in the context of the scene understanding and the agent positioning (Filliat & Mayer, 2003; Katsev, Yershova, Tovar, Ghrist, & LaValle, 2011; Muratet et al., 2005; Siagan, 2009; Sinopoli et al., 2001).

The mentioned tasks are difficult as they have to be solved online. During the robot flight it has to process information quickly to operate confidently in complex and sometimes changing environment. The computational power limitation of the board computer causes additional problems especially in the case of flying agents, where the mentioned limitations frequently are crucial (Tadeusiewicz, 2008). These problems could be solved efficiently if the possibility of application of GPU (Graphical Processing Unit), which enables us to process many tasks in parallel (Dally, 2010; Owens et al., 2007), is considered.

2. UAVs-the state of the art

UAVs, as devices with no pilot on board, can be remote controlled aircraft, e.g. flown by a pilot at a ground control station, or can fly autonomously, based on pre-programmed flight plans or more complex dynamic automation systems. UAVs are currently used for a number of missions, including reconnaissance and attack roles. As the capabilities grow for all types of UAVs, nations continue to subsidize their research and the development which leads to further advances and enables them to perform a multitude

☆ The paper was supported by the Polish Science Centre under grant no PBS1/A2/2/2012.

* Corresponding author. Tel.: +48 124162338.
E-mail addresses: azbielecki@gmail.com (A. Bielecki), tburatow@agh.edu.pl (T. Buratowski), smigielski.piotr@gmail.com (P. Śmigielski).

0957-4174/$ - see front matter © 2012 Elsevier Ltd. All rights reserved.
http://dx.doi.org/10.1016/j.eswa.2012.12.068

of missions. UAV not only perform intelligence, surveillance, and reconnaissance (ISR) missions any longer, but also this still remains their predominant type. Their roles have expanded to other areas. UAVs range in cost is from a few thousand dollars to tens of millions of dollars, and the aircraft used in these systems range in size from a Micro Air Vehicle (MAV) weighing less than one pound to large aircraft weighing over 18,000 lb. In order to be able to complete their task, UAVs are equipped with sensors that gather data from the environment which is then used in the control system. The objectives of these control systems are very different, like locating the drone in the space, or avoiding collisions. Different solutions with different characteristics are used depending on the mission. For example the Predator (Wilson, 2002), performing ISR missions is its objective. This can be achieved thanks to his long endurance (35 h) even when having a considerable size: 17 m wing span, 8 m in length, max takeoff weight 1157 kg, 147 kg of it is payload capacity. The ceiling of the studied model is over 7500 m and it can work autonomously for simple tasks and semi autonomously. The second one is a combat aircraft, the Arcturus T-20 (Yakimenko et al., 2009). The original idea was to design it for ISR missions too, but it was transformed into a combat aircraft. This model is much smaller than Predator, with 5.2 m wingspan, 2.8 m long body and a maximum payload of 80 kg. It also has a lower working range, around 90 km and 16 h endurance. Its ceiling is over 4500 m and it can work both autonomously and semi autonomously. The third example is also a combat model, but in this case, it is a helicopter, the MQ-8B Fire Scout (Downs et al., 2007). This is an example of autonomous solution, with a good lifting capacity, around 270 kg. On the other hand, the endurance of this model is lower than the ones in the previous examples, 8 h and it has a ceiling of 6000 m. The fourth design has a completely different application, transporting materials is its main task. It is the CQ-10 SnowGoose (Dalamagkidis, 2009). This small UAV is able to carry a payload of 75 lb with a maximum ceiling of almost 5500 m and a maximum endurance of 16 h, depending on the load the UAV is carrying. Our next example is one of the most controversial UAVs, the Nano hummingbird (Watts, Ambrosia, & Hinkley, 2012). This semi-autonomous UAV was created for surveillance and reconnaissance. The difference with the Predator example is clear, its size. This UAV has the size and form of a real hummingbird and can move at a maximum speed of 11 mph, being able to go forwards, backwards, sideways and rotate around its axis. The last example is the newest and more advanced solution for UAV, the HALEs (High altitude, long endurance), and one example is the Phantom Eye (Harrington, 2010). This model has been designed for gathering advanced intelligence and reconnaissance work, driven in combat conditions. It has a very large size, with a 46 m wingspan and it can carry up to 200 kg. The main advantages of this model are the almost 20,000 m service ceiling and its 4 days endurance, what makes this design suitable for new types of missions. As we have seen, there are diversified specifications for every kind of UAV depending on the mission and technology, but there are many problems to solve e.g. complexity of the cooperation: depending on how strong the cooperation is, faster or slower responses are necessary. There are also some barriers that prevent a higher investment, mainly in civil applications. The main one is that UAVs do not have an airspace authorization. But there are also technological problems like the lack of methodology to test the capacity of avoiding collisions (Shima & Rasmussen, 2009).

The mobile robot, used at AGH University of Science and Technology, is able to carry up to 10 kg of electronic systems and is used for tests. The robot is equipped with full airside and groundside UAV system components, the MP2128 MicroPilot's autopilot package with HORIZON software. This software enables the operator to monitor the autopilot, change waypoints, upload new flight plans, initiate holding patterns and adjust feedback loop, which is accessible during all the UAV flight. HORIZON Company, the producer of the MicroPilot, allows both the UAV developer and the end-user to access critical information in real-time. Up to eight user-defined sensors can be configured and displayed in three formats:

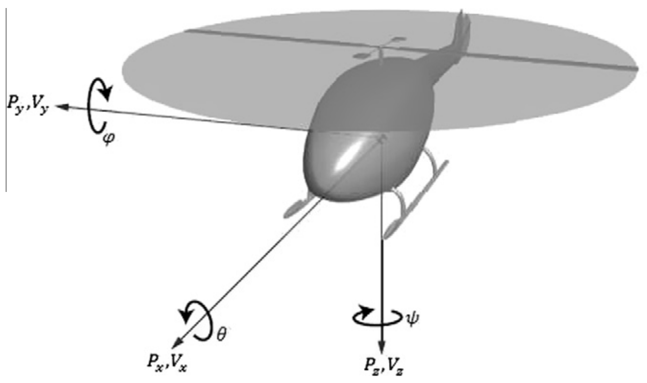

Fig. 1. Computational model of the UAV with assumed system of coordinates.

- Current Sensor values are displayed in an easy to read gauge format. Warning and danger levels can be set for each gauge.
- The Strip Chart graphs sensor specific variations over time.
- The Trace Route displays sensor data variations along the UAV's flight path.

Data from the autopilot are also recorded for post-flight analysis. The MicroPilot system can support four external analog data to digital converters (ADC). By using ADCs, you can display data from external sensors. Additionally it has analog input channels enhanced data collection capabilities for up to 32 channels. The sensors, for example, a three dimensional compass is useful when flying in high winds as it gives an accurate direction of the nose of the plane unlike the GPS heading which is the direction the plane is heading. For such operations it is necessary to describe a dynamic model of the system presented in Fig. 1.

3. Motivations

Research on the construction of on-board flight control systems, designed for unmanned aerial vehicle applications have been executed since the early 70s. The contemporary state of technology did not allow us for the development of autopilot to control the UAV based on a model helicopter. The development of MEMS (Micro Electro-Mechanical Systems) in later years helped to build gyroscopes, accelerometers and other sensors that were used in Inertial Navigation System-INS (Hong, Chun, Kwon, & Lee, 2008) with parameters suitable for UAV applications. The modern flight control system is a combination of several sensors, GNSS (Global Navigation Satellite System), software whose primary task is the estimation of the position, velocity and orientation. To make estimation of these parameters possible, several smaller systems providing information on these parameters of the flight should be integrated in one system. The first system, called Inertial Measurement Unit (IMU), is a combination of gyro sensors and accelerometers to measure angular velocity and acceleration, respectively around the axes X, Y, Z. The second one, named Attitude and Heading Reference System, is a combination of IMU with the Earth's magnetic field sensor that provides information about the direction of flight. The third system, called GNSS, provides information about the position, speed and altitude (Kaniewski, 2006; Kaniewski, 2007). The use of GNSS in the INS system can be implemented in two configurations as loosely coupled INS/GPS or tightly coupled INS/GPS. The difference between the configurations is that in the loosely coupled system configuration INS with GPS, for the visibility of less than four satellites, the most widely used Kalman filter is not able to estimate sufficiently good estimate of the position and speed of the system, while in tightly coupled INS/GPS configuration, even data from a single satellite is taken into account in the Kalman filter work process. Flight control systems can be categorized in two groups, the first group includes systems operating in open loop and the second one in a closed loop. The first group includes all the civilian uses, where the pilot always makes the final decision. The second group is a military application and use of UAVs, where the final decision always belongs the flight control system. On the basis of the presented data, concerning units used in UAV to set position and orientation, it can be noticed that it is significant to turn on the visual system for correcting the UAV position. Such attempts of using visual systems are made in SLAM (Simultaneous Localization and Mapping) technique, however, due to the complexity of calculations, they are difficult to interpret. On the other hand, the lack of using vision systems significantly influences lowering of technical abilities concerning UAV control as this forces the usage of the tightly coupled INS-GPS system, which is less exact and prone to external interruptions. The usage of the visual system supporting positioning and orientation. i.e., connected with INS-GPS via involving interpretation data in the process of Kalmans filter correction (Hide, Moore, & Smith, 2003; Sasiadek, Wang, & Zaremba, 2000), shall improve the estimation of the robots position in space. Additionally, the usage of the scene analysis shall enable it to interpret objects in a more exact way, which is a significant argument in favor of integrating the visual system with the control unit. Another important argument for applying the extended functionality of the visual system in UAV is the fact that it improves the autonomy of this type of units, which is very important from the point of view of UAVs application.

In this paper the problem of a vision system implementation for autonomous flying agents is considered in the context of industrial inspection tasks performed by UAVs. The construction industry is one of the most appropriate for robotization due to its current low level of automation (Balaguer, Giménez, Pastor, Padrón, & Abderrahim, 2000). Inspection operations of civil infrastructures such as buildings, bridges, building skeletons, complex roofs, factory chimneys, offshore platforms, wind turbines towers etc., are very important tasks (Balaguer et al., 2000; Balaguer, Giménez, & Jardon, 2005; Ciang, Lee, & Bang, 2008; Liu, Tang, & Jiang, 2010; Sattar, Rodriguez, & Bridge, 2009). On the one hand such constructions as elevated panels or long-span suspension bridges (Briassoulis, Mistriotis, & Giannoulis, 2010; Øiseth, Rönnquist, & Sigbjörnsson, 2010) are exposed to damages made by wind-caused vibrations, which are destructive according to the chaotic wind speed variability (Barszcz, Bielecka, Bielecki, & Wójcik, 2012). On the other hand they are inaccessible so flying agents are proper tools to inspect them. The high complexity of construction sites and environments must be taken into consideration in the operational aspects of the tasks performed by autonomous agents. The development of autonomous robots is very important from different points of view-safety of the infrastructure, safety of the human operators, quality of the inspections and increment of the periodicity of inspection (Balaguer et al., 2005). Aerial autonomous vehicles can be used effectively for such inspections (Metni & Hamel, 2007). The first part of the UAV's mission is the navigation from an initial position to a final position in an unknown three-dimensional environment. Then the robot has to recognize the object, that will be inspected, and understand the environment and the object spatial structure in the context of the mission, particularly in aspects of trajectory planning.

4. Problem formulation

Let us consider an UAV which is aimed at performing an inspection of an urban object-building, chimney, bridge, tower etc. The robot is autonomous one. It has to plan out the trajectory from the starting point to the mission area, then to analyze the mission scene-to locate the recognized object and to plan out the performing of the inspection task in the context of the structure of the investigated object and the scene properties. The agent's memory contains the map which is given a priori, for instance as a preprocessed satellite image of an urban environment. The robot is equipped with the mobile camera which is able to point to the ground in order to take the pictures of the surface that it flies above. The map that the robot carries presents the buildings extracted from the base satellite image (see Fig. 2). The problem of preprocessing which, beside other problems, consists of object extraction from the image, is out of the scope of this paper. By referring to the given two-dimensional map and the two-dimensional image from its senses-camera, radar-the agent orients itself in the environment, also by using other pieces of information, for instance GPS. More precisely, in order to find its location on the map the robot takes successive pictures of the ground below. Then it compares the extracted shape of the building from the picture and locate it in the bigger map. The example of the picture is shown in Fig. 2. The method of the objects recognition has to be rotation and scale invariant as the pictures of the ground are taken from different altitudes and various directions of the robot's flight. This problem belongs to the group of tasks that consist in recognition and representing polygonal-type objects by a robot sensory systems (Katsev et al., 2011). The method is extended with the recognition of the group of buildings. Then, the robot descends in order to explore the environment directly by using camera and sensors. This allows the agent to create the three-dimensional representation of the environment. Then, the environment should be understood and the desired object should be found. Thus, the studies can be divided in the following stages:

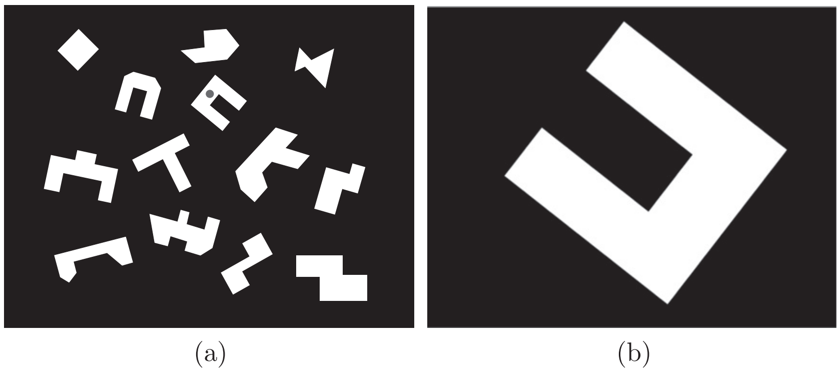

Fig. 2. (a) Preprocessed satellite map with extracted buildings. (b) The recognized object.

1. Creation a single, two-dimensional object representation, based on the camera image taken from high height.
2. Creation of two-dimensional scene representation based on the high height camera image. Then, the videoed scene should be recognized as a fragment of the previously given map. The single desired object should be recognized.
3. Creation of a representation of the three-dimensional scene, based on the worked out two-dimensional scene representation. The two-dimensional representation allows us to locate the buildings. When their localization is known, the robot can investigate individual objects in order to create their three-dimensional representation.
4. Understanding the three-dimensional scene.

The points 1 and 2 are the topics of this paper.

5. Syntactic algorithm of an two-dimensional object vectorization and recognition

In this section the algorithm based on syntactic methods for raster picture vectorization and object recognition is presented. The tests were carried out with the use of similar maps to these shown above in Fig. 2. In the big map one building that is supposed to be similar to the one from the small map is outlined. In a real test data the building was not outlined. To perform the action of objects recognition the new representation of the picture has to be introduced. Every extracted building is turned into a vector representation of its shape.

5.1. Vectorization algorithm

The vectorization algorithm works on prepared data based on the raster picture. It is a tabular, two dimension data in which 0 represents the background pixels and 1 represents the pixels of the building. This algorithm searches through matrix until it finds first unmarked building. It processes the shape of the building, turning it into vector representation and marking its space by changing values 1 into other number to avoid running into it while searching again for the next building to be vectorized. It is worth mentioning that each unmarked point that is found while searching through the matrix is a corner of a building or a point very close to the corner. Complex contour creating is the first step of vectorization of a single object. The contour of a building is found and its representation in the form of short vectors is created. Beginning from the first point of the contour (corner of a building) the next point is found using the *window*-a square used to define a sub-matrix. It is required for the *window* to have the size expressed with an odd value as it needs to have a central point. If starting point of the *window* lies outside the contour of a building the search of the border point is conducted clockwise; in the other case counter-clockwise-see (Bielecki, Buratowski, & Śmigielski, 2012) for details. The search conducted in such manner leads to circling around the building contour clockwise. It is important for the object recognition algorithm to build the list of points in the same direction in all cases.

Each encountered point in the border of processed building is turned from value 1 to 2. As a result, after collecting all the points around the building its border in the matrix representation consists of the sequence of groups of values 1 separated by 2. The length of groups depends on the size of the *window*. A presence of values 2 along the border is detected while searching for another building to determine if one has already been processed. It is accomplished in the following manner:

1. The algorithm iterates through matrix while the current value is 0 (the background).
2. If the value encountered is 2 (building already processed) iteration continues in the row of the matrix until 0 is reached again. Search for another building is continued then.
3. If the value encountered is equal to 1 the close surrounding of that point is checked for the presence of value 2. The surrounding is represented by the *window* with central point located in the point that is found on the border. The values in the *window* are checked. If value of 2 is found it indicates that the building was already processed. Then iteration in the row of the matrix is continued until 0 is reached and the algorithm searches for another building. If 2 is not present in the *window* it means that the building was not processed yet.

The output of building processing described in this step is the sequence of points which can be interpreted as a sequence of vectors located around the building contour (a point that is not the first or last in the sequence is the end of the one vector and the beginning of the next one).

After obtaining the long sequence of points in the contour it is vital to simplify that representation before running the object recognition algorithm. The algorithm of the final list obtaining is described in Bielecki et al. (2012). The results of the vectorisation algorithm for the data sets from Fig. 2 are presented in Fig. 3.

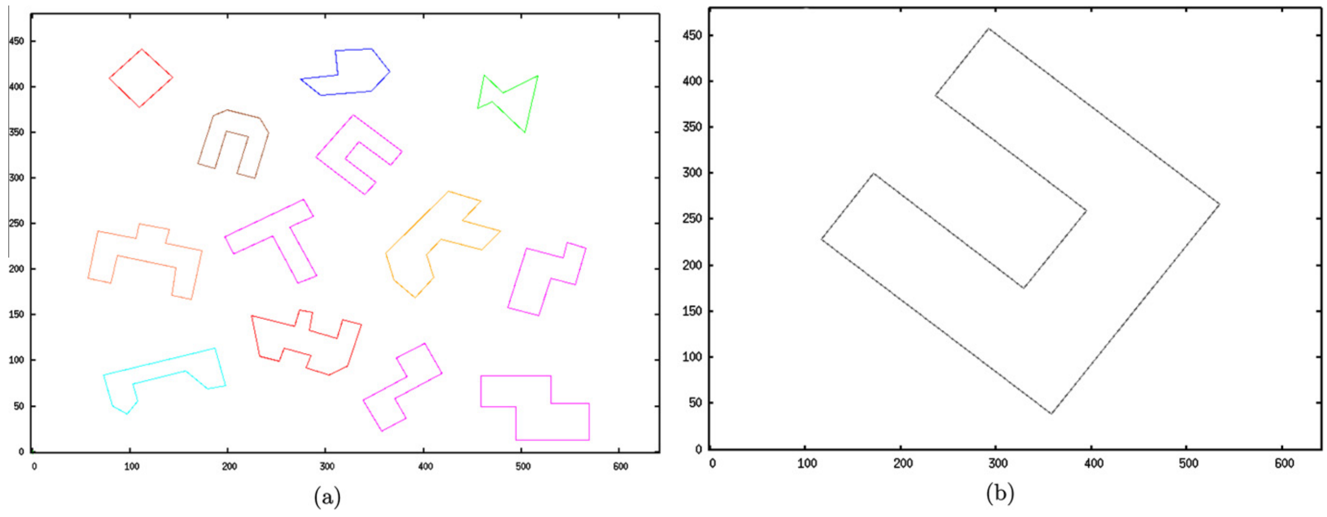

Fig. 3. (a) Vectorized map. (b) Vectorized picture of one building.

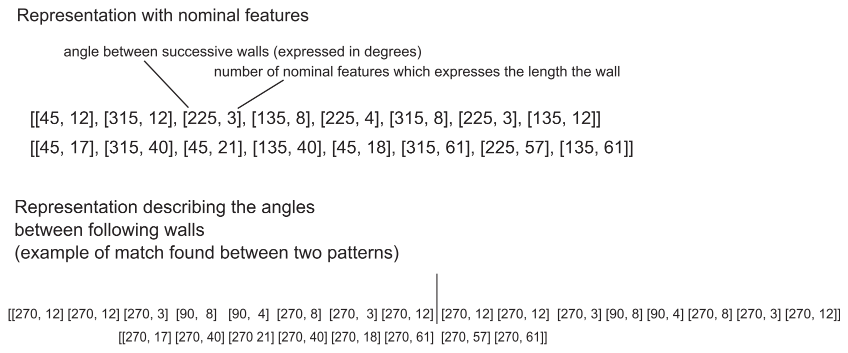

Fig. 4. Representaions used in object recognition.

5.2. Object recognition algorithm

The input for the object recognition algorithm consists of two vectorized maps. One is the map presenting large area filled with buildings and the other has only one building, similar to one placed in the big map, but scaled and rotated. The algorithm takes each vector representation of the building and compare it to the vector representation of the building from the small map. The comparison is conducted in the following way:

1. The number of corners of both buildings is compared. If it is different the buildings are not the same. Process is stopped. If it is the same the algorithm moves to another step.
2. The representation of the building which is the list of corners is turned into representation which uses angle and length of the walls. This can be considered as a nominal feature representation. To change the corner representation into nominal features each wall (pair of points representing the vector) of the building is taken and its length and angle is calculated. The nominal feature is described directly by the angle and the number of features is obtained by dividing the length of the wall by length of the feature. Finally each wall of the building is described by the pair (F, N), where F is the nominal feature sign (angle of the wall) and N is the number of features. Finally, as a result we obtain the list of form $((F_1, N_1), (F_2, N_2), \ldots, (F_m, N_m))$. This is the first form which needs to be transformed for further computations.
3. For both representations of the buildings that are compared a new list containing pairs of values is created. The first value expresses the angle between successive walls. The second value is the length of the wall preceding the corner (the number of nominal features that creates the wall). Actually the angle has the symbolic value as it is calculated as $\alpha_i = (F_{i+1} - F_i) \ mod$ 360; $i \in 1, \ldots, m-1$. The last angle (with apex in the first corner) is calculated as $\alpha_m = (F_1 - F_m) \ mod$ 360. That representation makes the difference in buildings rotation on the map irrelevant. The resulting list has the form: $((\alpha_1, N_1), (\alpha_2, N_2), \ldots, (\alpha_m, N_m))$.
4. Having such representation one list (representing one of the compared buildings) is doubled. Then the substring similar to the sequence of nominal features of the second building is searched in the doubled list (see Fig. 4). If the match is found the quotients of matching wall lengths is calculated. If all the quotients are similar (the standard deviation is below the given threshold) the buildings are treated as similar and the algorithm quits. If the quotients differ significantly (the lengths of the paired walls are different with respect to the scale) the search for the matching substring is continued. The maximum number of checks is equal to m.

6. Map recognition algorithm

The single object recognition algorithm constitutes the basics of the map recognition method. The aim of this method is to localize the submap in the overall picture. The submap consists of a group of buildings that is part of a big map. The method of a map recognition allows the picture to be scaled and rotated with relation to the big map. In further discussion, a center of an object (building) is treated as the center of mass of the vertex defined by the vector representation of a building. The algorithm can be divided into following parts:

6.1. Preparing the data structure

This step requires parsing both pictures and identifying all the buildings from the big map (B) that are similar to those from the submap (b). For each object b a list of matching objects from big map is created. There are also two values that describe relation between objects b and B which are obtained during single object recognition process. Those values are the scale and rotation. Scale is calculated as a quotient of matching wall lengths (which is similar for each walls pair when the match is found). Rotation is obtained as an angle between matching walls when the match is found. The final data structure is presented in Fig. 5.

6.2. Searching through data structure to find the matching objects

After parsing both maps and obtaining the proper data structure, algorithm iterates through the data structure in order to find the complete set of buildings located in the big map, that represent

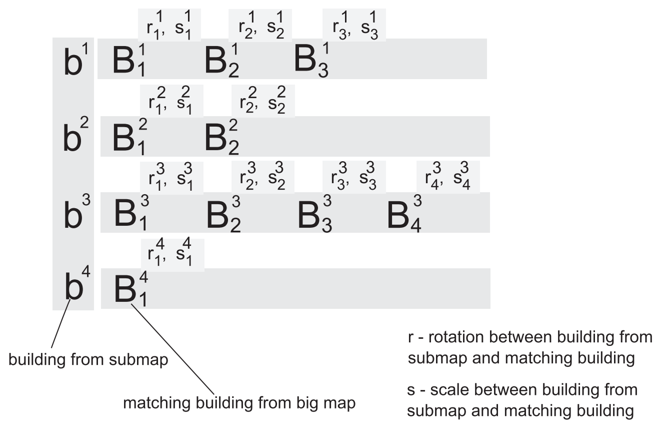
r_1^1, s_1^1 r_2^1, s_2^1 r_3^1, s_3^1
b^1 B_1^1 B_2^1 B_3^1
r_1^2, s_1^2 r_2^2, s_2^2
b^2 B_1^2 B_2^2
r_1^3, s_1^3 r_2^3, s_2^3 r_3^3, s_3^3 r_4^3, s_4^3
b^3 B_1^3 B_2^3 B_3^3 B_4^3
r_1^4, s_1^4
b^4 B_1^4
building from submap
matching building from big map
r - rotation between building from submap and matching building
s - scale between building from submap and matching building

Fig. 5. Data structure used in map recognition.

the relations between buildings from the submap. The first row in the data structure plays the role of the list of a *steering* objects. The attributes of a *steering* object (scale and rotation) will be searched in further steps among the rest of the lists. Once the *steering* object i is picked the next matching element is searched in the second list (the list with elements similar to the object b in Fig. 5). The element is matching when all conditions below are fulfilled (k-index of *steering* object in the first list, i-index of the list of similar objects that is actually processed, j-index of element in the list of similar objects):

1. $-\Delta r \leqslant r_k^1 - r_j^i \leqslant \Delta r$ - where Δr is the threshold of calculated rotations;
2. $-\Delta s \leqslant s_k^1 - s_j^i \leqslant \Delta s$ - where Δs is the threshold of calculated scales;
3. $-\Delta D \leqslant D(b^1, b^i) - s_j^i \times D\left(B_k^1, B_j^i\right) \leqslant \Delta D$ -where ΔD is the threshold of the distance between buildings, D (x, y) is a metric that, for a pair of buildings representations, returns an euclidean distance between their centers;

Of course the calculation of the third condition may be omitted when the scales in compared objects are different (the second condition). If the submap consists only of two buildings the above calculations are sufficient. Otherwise, starting from the third list of matching objects, another calculation is required. Its aim is to check if relations on the plane between objects in the submap are similar (correct to scale and rotation) to those selected from the big map. To accomplish this, two angles are calculated. First (α^1) is the one with the vertex in the center of building b^1 and sides determined by centers of buildings b^{i-1} and b^i. Second (α^2) has vertex in the center of building B_k^1 and sides determined by centers of buildings B_l^{i-1} and B_j^i (where l is an index of a building in the previous list for which match was found). If condition $-\Delta\alpha \leqslant \alpha^1 - \alpha^2 \leqslant \Delta\alpha$ (where $\Delta\alpha$ is the given threshold) is fulfilled it means that relations between three buildings on the submap and the big map are preserved and the match in step i is found. Fig. 6 shows which elements are chosen for the relation check.

The match between submap and the big map is found when on each list of matching elements at least one matching building is found. The search is stopped and the match between submap and the overall picture is not found when:

1. Any of the list with matching objects is empty (in this case the algorithm is finished without creating the data structure)
2. Or the algorithm encounters the first list where the match (with relation to the previously found objects) is not found.

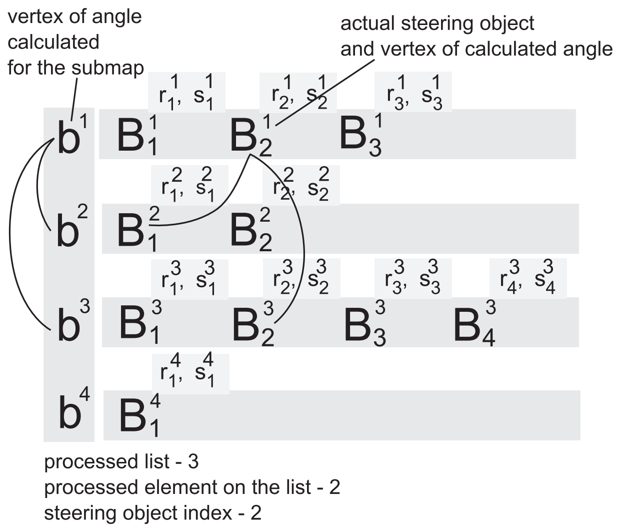

Fig. 6. Elements of the data structure which take part in calculation of angles.

7. Tests

This section is divided into two parts. First will show the capabilities of single object recognition method. The aim of the second section is to present the performance of the map recognition (multiple object recognition) algorithm. Each section is divided into two parts presenting results for two data sets. First is based on the set of artificial buildings which was mentioned above while discussing the algorithm (see Fig. 2). The second one is based on the satellite picture which was taken from GoogleMaps service (see Fig. 7a). The picture shows the area of Harvard College in Cambridge, Massachusetts. A group of buildings that were extracted for tests are highlighted. The actual map with extracted shapes is presented in Fig. 7b.

7.1. Single object recognition

In this section tests of the vectorisation and object recognition algorithm are presented. For the tests the algorithm parameters were taken as follows:

1. Window size - 9
2. Angle threshold T (for vectorisation) - 35°
3. Nominal feature length - 5

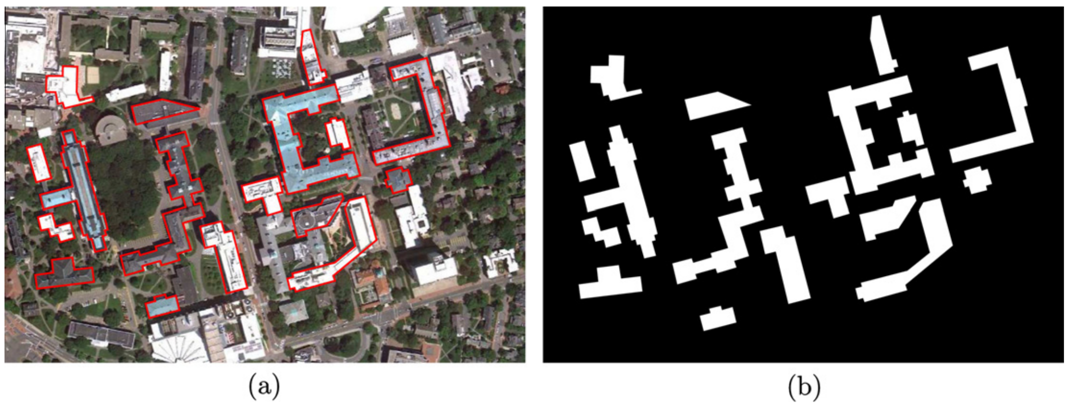

Fig. 7. (a) Satellite picture with marked buildings that were extracted. (b) Extracted buildings.

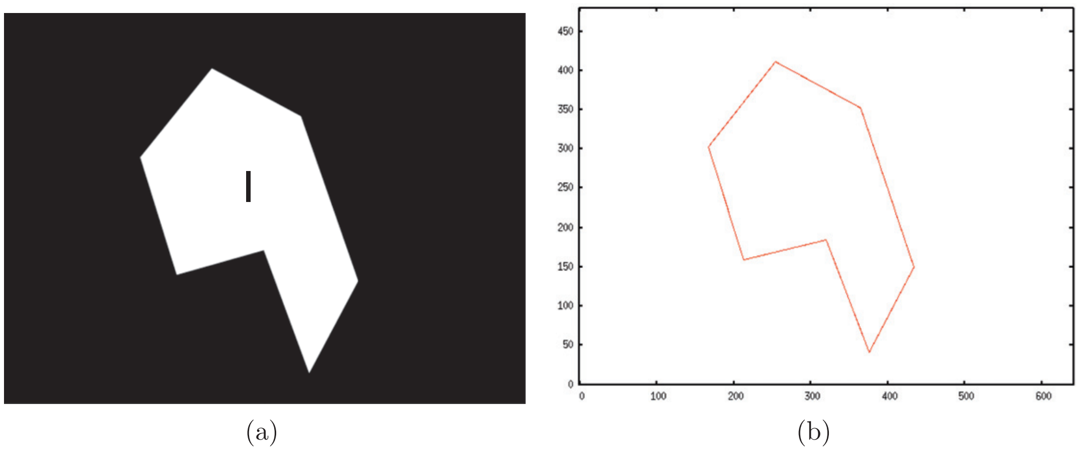

Fig. 8. Test case I, (a)-extracted image, (b)-vectorized image.

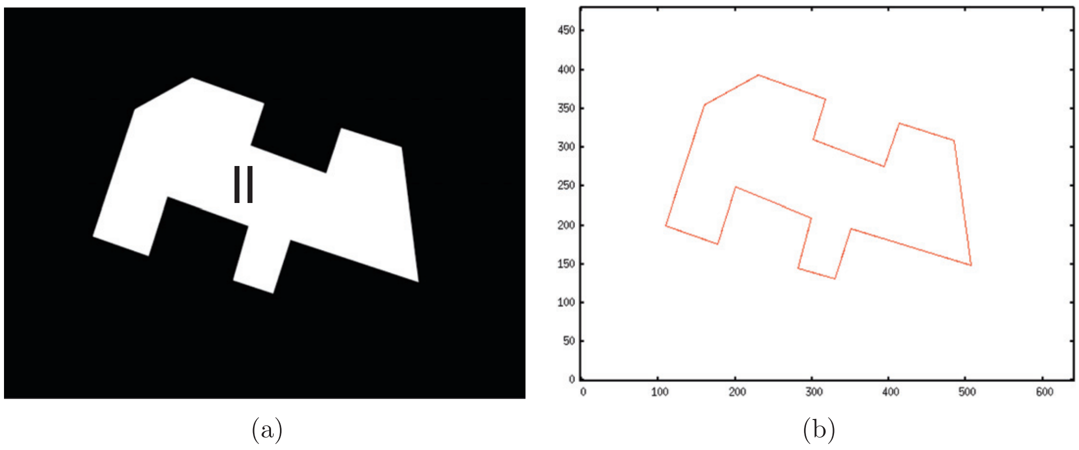

Fig. 9. Test case II, (a)-extracted image, (b)-vectorized image.

The artificial data set consists of two images (small maps) of buildings taken from the large picture (see Fig. 2). Their shapes were scaled and rotated. In Figs. 8 and 9 the extracted buildings are shown along with their vector representation obtained in the first step of the algorithm.

The Fig. 10 shows the extracted overall map with marked buildings for which the match was found. The matching buildings are marked with numbers I and II which correspond with test cases I and II presented above-see Figs. 8 and 9 respectively.

Let us turn to the test results based on real data. The testing data consists of four extracted images of buildings that can be found in Fig. 7. The Figs. 11–14 present both extracted images of buildings and their vector representation obtained after the first step of the algorithm.

The Fig. Fig. 15 shows the vectorized map of the extracted buildings obtained after vectorisation algorithm run and the map with extracted buildings. The buildings for which the match with test examples was found are marked with equivalent number.

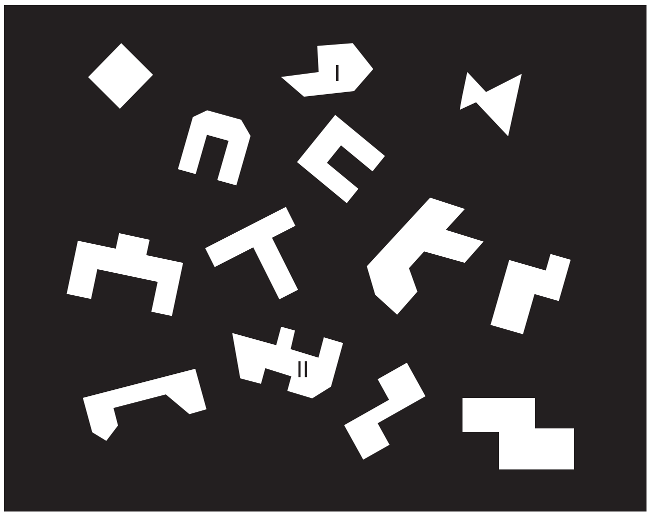

Fig. 10. Map with matching buildings.

7.2. Map recognition

In this part the test results of the map recognition algorithms are presented. For the tests the algorithm parameters were taken as follows:

1. Scale threshold Δs – 0.2
2. Distance threshold ΔD – 30 (value expressed in pixels)
3. Rotation threshold Δr – 6°
4. Threshold of angle between buildings $\Delta\alpha$ – 10°

The artificial data set consists of the image (see Fig. 16a) being the part of the large map-see Fig. 2. This small map shows the

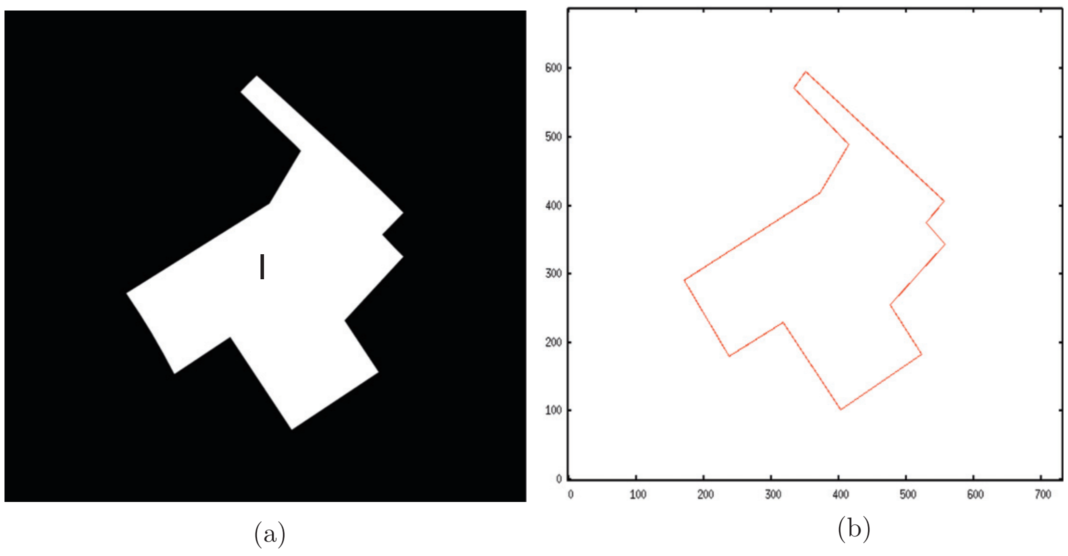

Fig. 11. Test case I, (a)-extracted image, (b)-vectorized image.

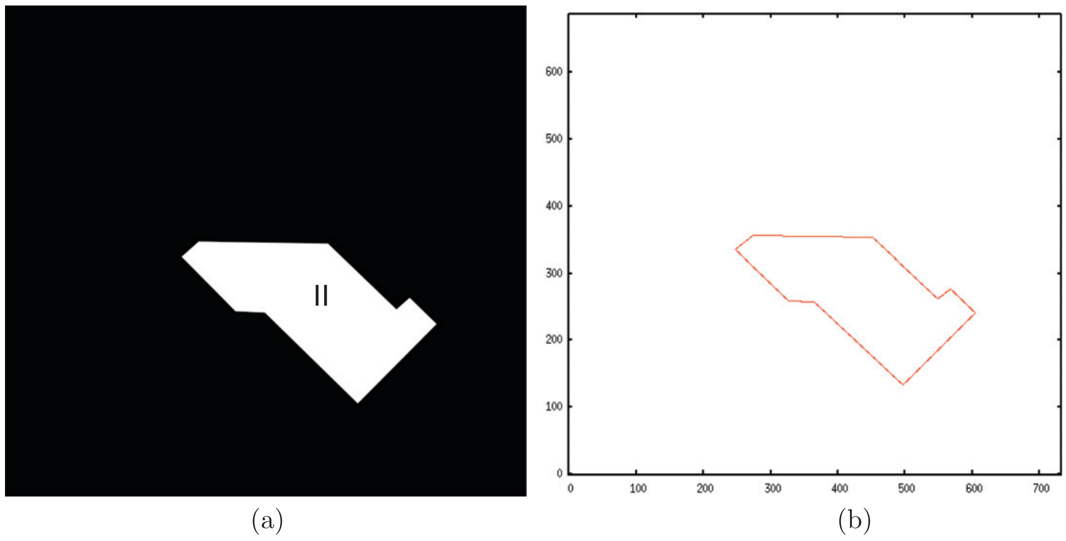

Fig. 12. Test case II, (a)-extracted image, (b)-vectorized image.

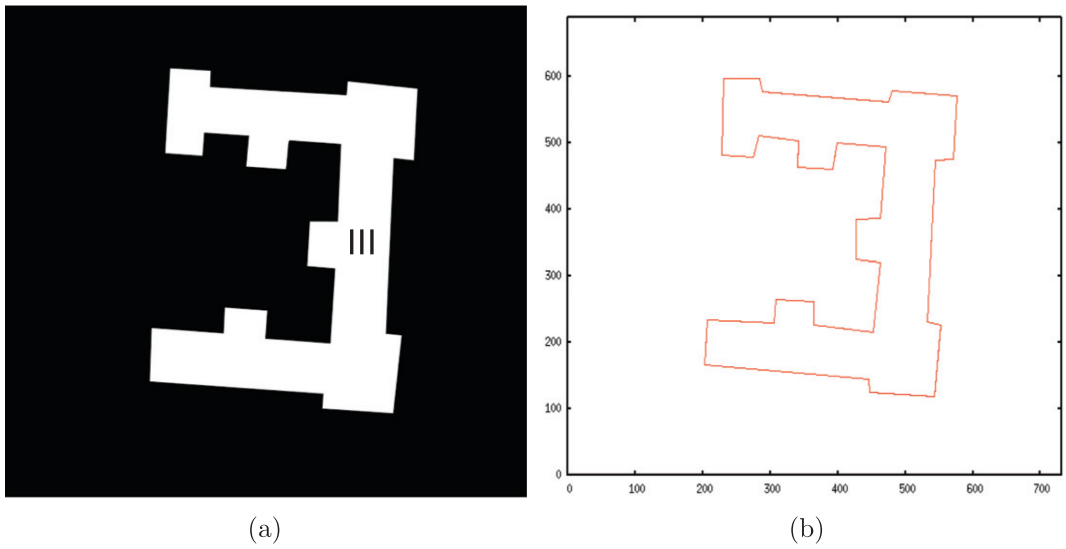

Fig. 13. Test case III, (a)-extracted image, (b)-vectorized image.

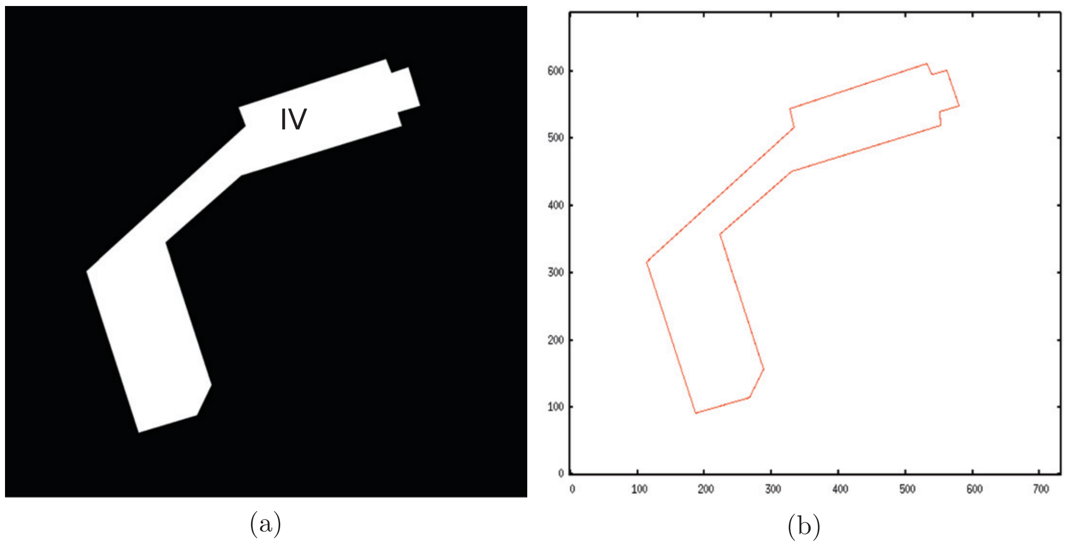

Fig. 14. Test case IV, (a)-extracted image, (b)-vectorized image.

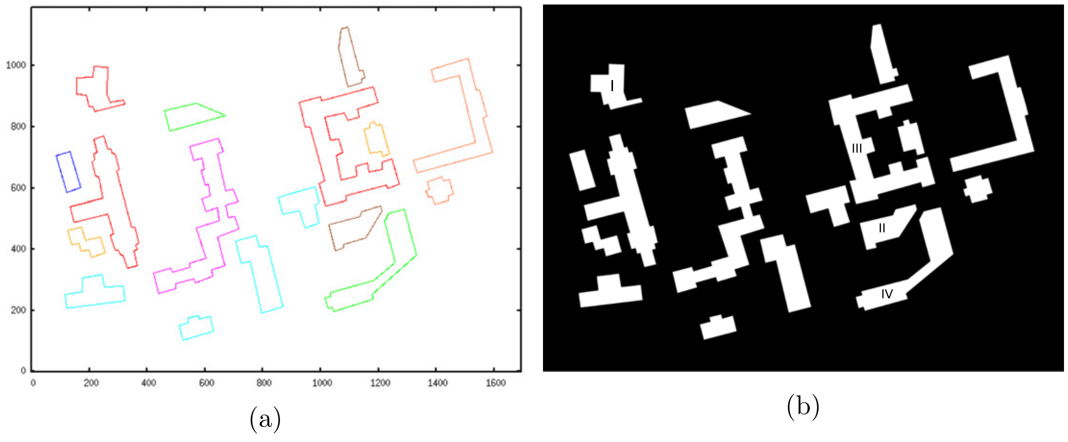

Fig. 15. (a) Vectorized map. (b) Map with extracted buildings.

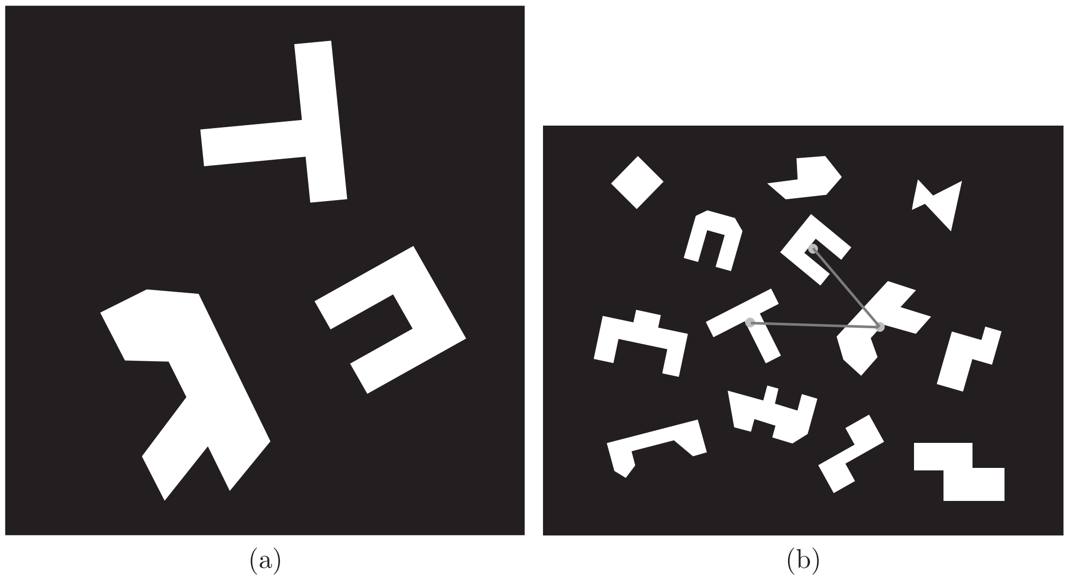

Fig. 16. (a) Small map presenting the group of buildings. (b) Full map presenting the group of buildings.

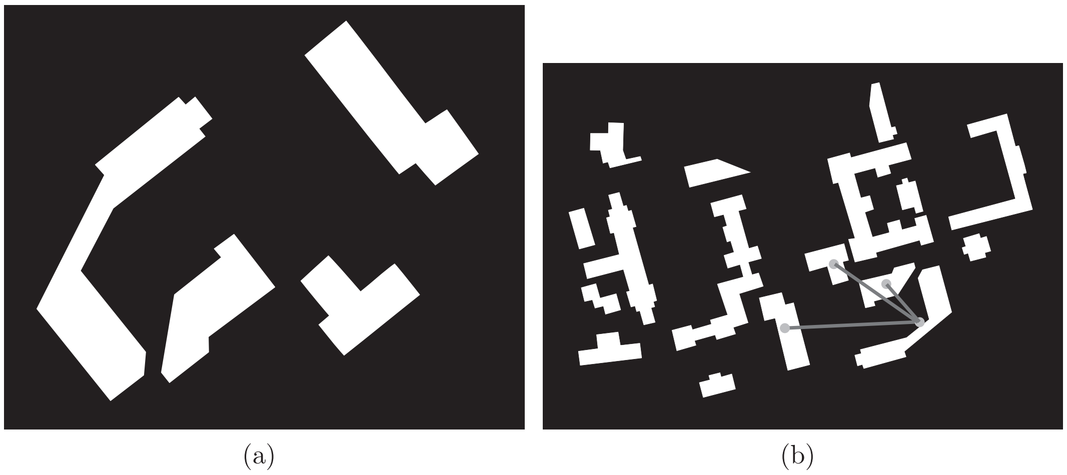

Fig. 17. (a) Small map presenting the group of buildings. (b) Full map presenting the group of buildings.

group of three extracted buildings. The group is scaled and rotated with relation to the large map.

In Fig. 16b the map with extracted buildings is presented again. The group of buildings for which the match with test example was found is marked with grey dots. The grey lines expresses the sides of the angles that took part in calculations of the relations between buildings.

The real data set consists of the image (see Fig. 17a) which is the part of the large map of Harvard College (see Fig. 7b). There is a group of four adjacent buildings in the small map. The group is scaled and rotated with relation to the large map.

In Fig. 17 the map with extracted buildings is presented again. The group of buildings for which the match with test example was found is marked with grey dots. The grey lines expresses the sides of the angles that took part in calculations of the relations between buildings.

8. Concluding remarks

In this paper the algorithm, based on syntactic methods for raster picture vectorization and single object recognition, has been presented. The algorithm of recognition of a two-dimensional scene consisting of such object has been introduced as well. Both the algorithms turned out to be effective, which has been pointed out by using both the created artificial objects and maps and the maps, based on satellite images. In particular the algorithm are insensitive to scale changing and the map rotations.

It should be stressed that the obtained results, including the performed experiments, have only computer scientific aspect-the software has not been implemented on an embodied agent. The three-dimensional scene representation is necessary to test an UAV equipped with the module of vision analysis.

So far only polygonal shapes have been taken into account in the part of studies concerning two-dimensional scene-stages 1 and 2 in the schedule presented in Section 4. In the closer perspective, apart from the realization of the points 3 and 4 (see Section 4), the class of objects consisting of both line segments and arcs should be taken into consideration. In such a case the shape languages should be a good starting point for creating the system for the scene analysis and understanding- (Jakubowski, 1982; Jakubowski, 1985; Jakubowski, 1986; Jakubowski, 1990). This formalism seems to be useful also for creation the three-dimensional scene representation-see (Jakubowski & Flasiński, 1992).

In the further perspective fully autonomous robots are planned to be used for the unknown environment exploration. The inspection of industrial and urban objects is a relatively simple task in the context of the scene analysis. If the robot is used for exploring un unknown natural environment, for instance, the Moon or planets (Pederson, Kortencamp, Wettergreen, & Nourbakhsh, 2003), the agent's scene understanding capabilities have to go far beyond syntactic scene analysis. In a such case the agent should be equipped with cognitive abilities. Cybernetic models of human cognitive structures (Bielecki, Kokoszka, & Holas, 2000; Kokoszka, Bielecki, & Holas, 2001) and models of the nervous system, that are their support (Barlow, 1999;Bielecki, 2001; Bielecki, Jabłoński, & Kędzierski, 2004;Bielecki & Kalita, 2008; Bielecki & Kalita, 2012;Bielecki, Kalita, Lewandowski, & Siwek, 2010; Bielecki, Kalita, Lewandowski, & Skomorowski, 2008;Bielecki & Ombach, 2004; Bielecki & Ombach, 2011;Fiori, 2006), seem to be a good starting point for studies of agent's cognitive module based on its vision system.

References

Balaguer, C., Giménez, A., Pastor, J. M., Padrón, V. M., & Abderrahim, M. (2000). A climbing autonomous robot for inspection applications in 3D complex environments. *Robotica, 18*, 197–287.

Balaguer, C., Giménez, A., & Jardon, A. (2005). Climbing robots mobility for inspection and maintenance of 3D complex environments. *Autonomous Robots, 18*, 157–169.

Barszcz, T., Bielecka, M., Bielecki, A., & Wójcik, M. (2012). Wind speed modelling using Weierstrass function fitted by a genetic algorithm. *Journal of Wind Engineering and Industrial Aerodynamics, 109*, 68–78.

Barlow, H. B. (1999). What is the computational goal of neurocortex? In C. Koch & J. L. Davis (Eds.), *Large-scale neuronal theories of the brain*. Cambridge: MIT Press.

Bielecka, M., Skomorowski, M., & Bielecki, A. (2007). Fuzzy syntactic approach to pattern recognition and scene analysis. In *Proceedings of the 4th international conference on informatics in control, automatics and robotics ICINCO07, ICSO intelligent control systems and optimization, robotics and automation* (Vol. 1, pp. 29–35).

Bielecka, M., Bielecki, A., Korkosz, M., Skomorowski, M., Wojciechowski, W., & Zieliński, B. (2010). Application of shape description methodology to hand radiographs interpretation. *Lecture Notes in Computer Science, 6374*, 11–18.

Bielecka, M., Bielecki, A., Korkosz, M., Skomorowski, M., Wojciechowski, W., & Zieliński, B. (2011). Modified Jakubowski shape transducer for detecting osteophytes and erosions in finger joints. *Lecture Notes in Computer Science, 6594*, 147–155.

Bielecki, A. (2001). Dynamical properties of learning process of weakly nonlinear and nonlinear neurons. *Nonlinear Analysis: Real World Applications, 2*, 249–258.

Bielecki, A., Buratowski, T., & Śmigielski, P. (2012). Syntactic algorithm for two-dimensional scene analysis for unmanned flying vehicles. *Lecture Notes in Computer Science, 7594*, 304–312.

Bielecki, A., Jabłoński, D., & Kędzierski, M. (2004). Properties and applications of weakly nonlinear neurons. *Journal of Computational and Applied Mathematics, 164-165*, 93–106.

Bielecki, A., & Kalita, P. (2008). Model of neurotransmitter fast transport in axon terminal of presynaptic neuron. *Journal of Mathematical Biology, 56*, 559–576.

Bielecki, A., & Kalita, P. (2012). Dynamical properties of the reaction–diffusion type model of fast synaptic transport. *Journal of Mathematical Analysis and Applications, 393*, 329–340.

Bielecki, A., Kalita, P., Lewandowski, M., & Siwek, B. (2010). Numerical simulation for neurotransmitter transport model in axon terminal of presynaptic neuron. *Biological Cybernetics, 102*, 489–501.

Bielecki, A., Kalita, P., Lewandowski, M., & Skomorowski, M. (2008). Compartment model of neuropeptide synaptic transport with impulse control. *Biological Cybernetics, 99*, 443–458.

Bielecki, A., Kokoszka, A., & Holas, P. (2000). Proposals of mathematical model of consciousness basing on dynamical systems theory. *International Journal of Neuroscience, 104*, 29–47.

Bielecki, A., & Ombach, J. (2004). Shadowing property in analysis of neural network dynamics. *Journal of Computational and Applied Mathematics, 164–165*, 107–115.

Bielecki, A., & Ombach, M. (2011). Dynamical properties of a perceptron learning process-structural stability under numerics and shadowing. *Journal of Nonlinear Science, 21*, 579–593.

Bonin-Font, F., Ortiz, A., & Oliver, G. (2008). Visual navigation for mobile robots: A survey. *Journal of Intelligent and Robotic Systems, 53*, 263–296.

Briassoulis, D., Mistriotis, A., & Giannoulis, A. (2010). Wind forces on porous elevated panels. *Journal of Wind Engineering and Industrial Aerodynamics, 98*, 919–928.

Ciang, C. C., Lee, J. R., & Bang, H. J. (2008). Structural health monitoring for a wind turbine system: A review of damage detection methods. *Measurement Science and Technology, 19*(12), 1–20.

Dalamagkidis, K. (2009). Aviation history and UAS. *Intelligent Systems, Control and Automation: Science and Engineering, 36*, 9–28.

Dally, W. J. (2010). The GPU computing era. *IEEE Micro, 30*(2), 56–69.

Downs, J., Prentice, R., Dalzell, S., Busachio, A., Ivler, C. M., Tischler, M. B., & Mansur, M. H. (2007). Control system development and flight test experience with the MQ-8B Fire Scout vertical take-off unmanned aerial vehicle. In *Presented at American Helicopter Society 63rd annual forum and technology display, Virginia Beach, May 1–3*.

Filliat, D., & Mayer, J. A. (2003). Map-based navigation in mobile robots. A review of localization strategies. *Journal of Cognitive Systems Research, 4*, 243–283.

Fiori, S. (2006). A theory for learning by weight flow on Stiefel–Grassman manifold. *Neural Computation, 13*, 1625–1647.

Flasiński, M. (1988). Parsing of edNLC-graph grammars for scene analysis. *Pattern Recognition, 21*, 623–629.

Flasiński, M. (1993). On the parsing of deterministic graph languages for syntactic pattern recognition. *Pattern Recognition, 26*, 1–16.

Harrington, C. (2010). Boeing UAVs gearing up for first test-flights. *Jane's Defence Weekly, 47*(30), 10–28.

Hide, C., Moore, T., & Smith, M. (2003). Adaptive Kalman filtering for low-cost INS/GPS. *Journal of Navigation, 56*, 143–152.

Hrabar, S., Sukhatme, G. S., Corke, P., Usher, K., & Roberts, J. (2005). Combined optic-flow and stereo-based navigation of urban canyons for a UAV. In *Proceedings of the international conference on intelligent robots and systems IROS'2005* (pp. 3309–3316).

Hong, S., Chun, H., Kwon, S., & Lee, M. H. (2008). Observability measures and their application to GPS/INS. *IEEE Transactions on Vehicular Technology, 57*, 97–106.

Jakubowski, R. (1982). Syntactic characterization of machine parts shapes. *Cybernetics and Systems, 13*, 1–24.

Jakubowski, R. (1985). Extraction of shape features for syntactic recognition of mechanical parts. *IEEE Transactions on Systems, Man and Cybernetics, 15*, 642–651.

Jakubowski, R. (1986). A structural representation of shape and its features. *Information Sciences, 39*, 129–151.

Jakubowski, R. (1990). Decomposition of complex shapes for the structural recognition. *Information Sciences, 50*, 35–71.

Jakubowski, R., & Flasiński, M. (1992). Towards a generalized sweeping model for designing with extraction and recognition of 3D solids. *Journal of Design and Manufacturing, 2*, 239–258.

Kaniewski, P. (2006). Aircraft positioning with INS/GNSS integrated system. *Molecular and Quantum Acoustics, 27*, 149–168.

Kaniewski, P. (2007). Closed loop INS/GNSS integrated positioning system. *Molecular and Quantum Acoustics, 28*, 151–163.

Katsev, M., Yershova, A., Tovar, B., Ghrist, R., & LaValle, S. M. (2011). Mapping and pursuit-evasion strategies for a simple wall-following robot. *IEEE Transactions on Robotics, 27*, 113–128.

Kokoszka, A., Bielecki, A., & Holas, P. (2001). Mental organization according to metabolism of information and its mathematical description. *International Journal of Neuroscience, 107*, 173–184.

Liu, W., Tang, B., & Jiang, Y. (2010). Status and problems of wind turbine structural health monitoring techniques in China. *Renewable Energy, 35*, 1414–1418.

Metni, M., & Hamel, T. (2007). A UAV for bridge inspection: Visual servoing control law with orientation limits. *Automation in Construction, 17*, 3–10.

Muratet, L., Doncieux, S., Briere, Y., & Meyer, J. A. (2005). A contribution to vision-based autonomous helicopter flight in urban environments. *Robotics and Autonomous Systems, 50*, 195–229.

Øiseth, O., Rönnquist, A., & Sigbjörnsson, R. (2010). Simplified prediction of wind-induced response and stability limit of slender long-span suspension bridges, based on modified quasi-steady theory: A case study. *Journal of Wind Engineering and Industrial Aerodynamics, 98*, 730741.

Owens, J. D., Luebke, D., Govindaraju, N., Harris, M., Krüger, J., Lefohn, A. E., et al. (2007). A survey of general-purpose computation on graphics hardware. *Computer Graphics Forum, 26*, 80–113.

Pederson, L., Kortencamp, D., Wettergreen, D., & Nourbakhsh, I. (2003). A survey of space robotics. In *Proceedings of the 7th international symposium on artificial intelligence, robotics, and automation in space, Munich, Germany*.

Sasiadek, J. Z., Wang, Q., & Zaremba, M. B. (2000). Fuzzy adaptive Kalman filtering for INS/GPS data fuzion. In *Proceedings of the IEEE international symposium on intelligent control* (pp. 181–186).

Sattar, T., Rodriguez, H., & Bridge, B. (2009). Climbing ring robot for inspection of offshore wind turbines. *Industrial Robot: An International Journal, 36*, 326–330.

Shima, T., & Rasmussen, S. (Eds.). (2009). *UAV cooperative decision and control: Challenges and practical approaches. Advances in design and control*. SIAM.

Siagan, C. (2009). Biologically inspired mobile robot vision localization. *IEEE Transactions on Robotics, 25*, 861–873.

Sinopoli, B., Micheli, M., Donato, G., & Koo, T. J. (2001). Vision based navigation for an unmanned aerial vehicle. In *Proceedings of the international conference on robotics and automation ICRA* (Vol. 2, pp. 1757–1764).

Skomorowski, M. (1998). Parsing of random graphs for scene analysis. *Machine Graphics and Vision, 7*, 313–323.

Skomorowski, M. (1999). Use of random graph parsing for scene labeling by probabilistic relaxation. *Pattern Recognition Letters, 20*, 949–956.

Skomorowski, M. (2007). Syntactic recognition of distorted patterns by means of random graph parsing. *Pattern Recognition Letters, 28*, 572–581.

Tadeusiewicz, R. (1992). *Vision systems of industrial robots*. Warszawa: WNT [in Polish].

Tadeusiewicz, R. (2008). A visual navigation system for a mobile robot with limited computational requirements. *Problemy Eksploatacji, 4*, 205–218.

Tadeusiewicz, R. (2004). *Medical image understanding technology*. Berlin, Heidelberg: Springer.

Tadeusiewicz, R. (2006). Automatic image understanding-a new paradigm for intelligent medical image analysis. *Bio-Algorithms and Med-Systems, 2*(3), 3–9.

Tadeusiewicz, R., & Flasiński, M. (1991). *Pattern recognition*. Warsaw: Polish Scientific Publishers, PWN [in Polish].

Watts, A. C., Ambrosia, V. G., & Hinkley, E. A. (2012). Unmanned aircraft systems in remote sensing and scientific research: Classification and considerations of use. *Remote Sensing, 4*, 1671–1692.

Wilson, J. R. (2002). UAVs and the human factor. *Aerospace America Magazine, 40*, 54–57.

Yakimenko, O. A., Slegers, N. J., Bourakov, E. A., Hewgley, C. W., Bordetsky, A. B., Jensen, R. P., Robinson A. B., Malone J. R., & Heidt, P. E. (2009). Mobile system for precise aero delivery with global reach network capability. In *Proceedings of the 7th IEEE international conference on control and automation, Christchurch, New Zealand* (pp. 1394–1398).