**Waldemar WÓJCIK[1], Orken MAMYRBAYEV[2], Ainur AKHMEDIYAROVA[3], Dinara KASSYMOVA[3], Irbulat UTEPBERGENOV[2]**

Lublin University of Technology, Poland (1), Institute of Information and Computational Technologies, Kazakhstan (2), Kazakh National Research Technical University after K.I.Satpaev (3)

doi:10.15199/48.2017.06.31

# Problem of placement of the minimal number of cameras at a given transport network

***Abstract**. This article discusses the planning and placement of the surveillance system in the existing urban transport network. The result is the algorithms and programs that minimize the number of points where video cameras are placed in a given area for its complete traceability. The work may be used in the design of intelligent transport systems in the major cities.*

***Streszczenie.** W artykule omówiono planowanie rozmieszczenia kamer systemu nadzoru w istniejącej sieci transportu miejskiego. Rezultatem są algorytmy i programy, które minimalizują liczbę punktów, w których kamery wideo są umieszczane w danym obszarze dla jego pełnej identyfikowalności. Prace mogą być wykorzystane przy projektowaniu inteligentnych systemów transportowych w dużych miastach. (**Problem rozmieszczenia minimalnej liczby kamer w danej sieci transportowej)***

**Keywords:** problem of minimal surface, Balash's additive algorithm.
**Słowa kluczowe:** Problem minimalnej powierzchni, algorytm addycyjny Balasha.

## Introduction

Traffic jam is one of the most pressing problems of modern cities with supersaturated roads. Shortness of movement has become one of the essential features of big city life. Therefore, introduction of intelligent transport systems (ITS) is of interest for almost every industrialized city [1]. Such systems are designed to increase the capacity of streets and highways, to improve communication speed and traffic safety, to reduce delays at intersections, to reduce the consumption of fuel and lubricants, to improve environment. The basis of all intelligent transport systems is continuous collection of information about traffic conditions, flow rates, accidents and conditions for movement of cars, which is further processed and conveniently brought to the attention of drivers, or used for traffic control [1,2].

Video camera is one of the most simple, familiar and effective tools used in monitoring. Its advantage over other tracking systems is in capability to see directly everything that is in the camera's view. Modern specialized software for processing images from cameras is capable to measure the parameters of the moving stream (e.g., the vehicles movement speed, traffic flow density), to determine the state registration plates of vehicles, to detect violations of traffic rules [3]. Subsequently, this information may also be used for various purposes.

Development of digital technology and cheapening of video cameras caused an opportunity to organize video surveillance (with the subsequent development of ITS) in cities with a minimum number of cameras providing the most complete monitoring of the city's transport network. This work involves the implementation of this particular approach.

## Preliminary analysis

In order to build an adequate model it requires some initial information about problem areas in the city, the speed limit, the average occupancy of individual sites, dependence on the time of the day and on season. Also, it requires the city map and knowledge of the distance from one point to another.

In the first step, the city area is visually divided into the areas with dense road network, connected with other areas by main roads. Cameras location in one area will not affect their placement in another area. Thus, the problem is rather simplified, as it is divided into several sub-tasks of smaller dimension.

In the same way, it is possible to divide the areas into subareas. The next step is to determine the roads in the area, where the bulk of the traffic flow and emergency sections is concentrated. The roads which do not contribute substantially to the flow should be excluded from consideration. These include roads with a low load, in which the occurrence of emergencies does not change the nature of the motion, and internal roads of residential areas. The areas directly adjacent to the highways should not be excluded.

The next task will be to define the possible camera locations in the obtained territory. It is convenient to arrange them in the way that makes possible to review as many traced areas as possible. These points are mainly intersections, turns, roofs or corners of buildings, high-rise buildings. After defining the main points of placement, it is necessary to divide the entire required territory into the sets of areas viewed by each camera, and to add or remove cameras when necessary [4].

Let's consider the three possible situations: partial tracking, partial overlap and rounded areas.

1. Partial tracking

This situation is typical for long distances between road junctions, when cameras cover part of the site. If it is required to track section along the entire length, it is necessary to add points to the camera placement, depending on the area length. The easiest option in this case is to place cameras over a distance equal to twice the radius of the camera view (or one radius, if the camera cannot perform a circular rotation).

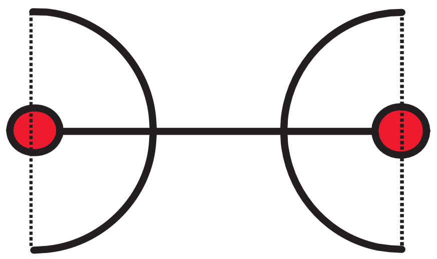
Fig.1. Partial tracking scheme

If this site is not dangerous or it does not require tracking over the entire length, then two cameras will be sufficient, because it is possible to assess the emergency situation that arises in the interval by secondary signs (for example, by comparing the flux density at the ends of the gap).

In non-emergency long sections of highways, which crosses only secondary roads, it also does not require the full tracking and camera conveniently placed either in the turns or through a certain distance, which can be much larger than the radius of the camera's view.

2. Partial overlap

The same section may be traced to varying degrees by means of two or more cameras. At insignificant overlapping (for example, insignificance of overlap may be determined in a percentage, or in units of length, and its specific value depends on the considered area), it is possible to divide the covered area into the desired number of subareas and to indicate the observed territory for each camera. If the overlap is significant, we can assume that both the camera trace the entire area. Examples of such situations are shown in Figure 2.

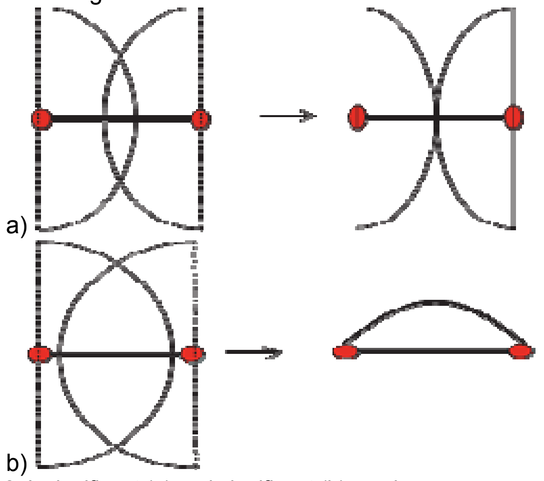

Fig.2. Insignificant (a) and significant (b) overlap

3. Rounded areas

The rounded sections of the road are traced either by oscillation or by a point in rounding, which allows to monitor the entire turn (Figure 3) without difficulty.

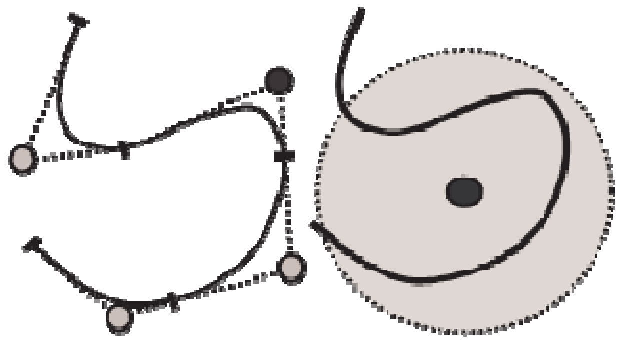
Fig. 3. Rounding

Also, the entire set of possible placement points you want to select those in which the camera should be mandatory. Such points may be, for example, bridges, tunnels, especially emergency areas.

**Mathematical formulation of the problem**

After the analysis, the problem is formulated as follows. Let $I=\{1,...m\}$ is a set of all possible locations of the cameras;

$J=\{1,...n\}$ is a set of all observed sections;

$$a_{ij}=\begin{cases}1, & \text{if camera i monitors the area}\\ 0, & \text{otherwise}\end{cases}$$

Variable tasks:

$$x_i=\begin{cases}1, & \text{if camera is placed in area i}\\ 0, & \text{otherwise}\end{cases}$$

Then the mathematical model of the problem to be solved looks like

$$\min\sum_{i\in I}x_i$$

under constraints: $\sum_{i\in I}a_{ij}x_i\ge 1$, $j\in J$ and $x_i\in\{0,1\}$, $i\in I$.

This problem is known in integer programming as the problem of minimal surfaces (PMS) [5,6].

**3. Algorithms for computing**

In order to solve PMS, a large number of both exact accurate and approximation methods of solution is developed. All existing accurate algorithms are associated with search within the finite set given by initial data. By volume of such search as a function of the number of source data, the accurate algorithms for PMS solution are exponential, which makes them impossible to use for problems with the dimension of the matrix over 100. Generally, approximation algorithms are used for more practical solutions of real problems, where leaders are so called «greedy» algorithms [7]. The idea of this type of algorithm is associated with sequential construction of valid options. At each algorithmic stage the attempt is made to achieve a certain local auxiliary target (minimizing the number of outstanding constraints, optimization of the initial target function, etc.) using the selected parameters at this stage. Local approach of such type does not provide any attempts to assess neither previous, nor further steps of the algorithm.

In this problem, it can be expected that results of the preliminary analysis will allow to construct the input data in such a way that the problem can be solved by means of accurate algorithms in a reasonable time. Let's use two algorithms to solve the problem: Balash's additive algorithm and greedy algorithm proposed by D.S. Johnson [6].

*Balash's additive algorithm.* The algorithm is presented in [2], it is a method of implicit enumeration, in general, it is designed to solve the following problem:

$$Z(x)=\sum_{j=1}^{n}c_jx_j\to\min_{x\in B^n}$$

$$\sum_{j=1}^{n}a_{ij}x_j\ge b_i,\ i=1,...,m.$$

Let a $\{x_j:x_j=1, j\in N_1; x_j=0, j\in N|N_1, N=\{1,...,n\}\}$ set be a solution.

The solution is valid if the following inequality is true:

$$\sum_{j=1}^{n}a_{ij}x_j\ge b_i\cdot$$

Let's suppose that the inequalities $0\le c_1\le c_2\le...\le c_n$ are valid. The problem has $2^n$ different solutions. We divide the set of all solutions into $(n+1)$ subsets with indexes $k=0,1,\ldots,n,$ so that $k$-th subset contains all the solutions with $k$ variables, equal to one, and $n$–$k$ variables, equal to zero. Consequently:

- when $k=0$, the subset of solutions consists of a single solution $x=0$;

- when $k = 1$, the subset of solutions includes $C_n^1$ solutions where $x_i = 1$, $x_j = 0$, $j \neq i$, $i = 1,...,n$;
- $k$-th subset consists of $C_n^k$ solutions.

Then, all the solutions are ordered in the graph, where each vertex containing the list $N_1$, presents a solution in which the variables with indices are equal to one, and the rest ones are equal to zero.

If there is a path in the graph from the vertex $u$ to vertex $v$, the vertex $u$ is called preceding to the vertex $v$, and the vertex $v$ is called succeeding vertex $u$. Thus, the solutions are partially ordered.

For the validation of solutions, let's check the condition $\sum_{j=1}^{n} a_{ij}x_j \geq b_i$ .

The algorithm starts from the vertex $x = 0$, then looks through the succeeding vertices. If a vertex corresponds to an admissible solution, then no need to watch the succeeding vertices, they are excluded.

Three options are developed on the basis of Balash's algorithm: approximation algorithm C1, Climb algorithm – when the chart pass below and Descend Algorithm – when the chart pass above. Programs for the algorithms implementation are written in programming language C.

1) approximation algorithm C1. The algorithm uses set-theoretic formulation of the problem and is presented in [4].

The sets $S_i$ are built for our problem as follows: for each index $i$ set $S_i$ is a set of indices of those sites, which can be traced by $i$-th camera.

Let $Sub$ contains the covered subsets and $Un\mathrm{cov}$ contains elements which are not covered yet. The C1 algorithm is as follows:

1. Let's assume $Sub = \varnothing$, $Un\mathrm{cov} = \bigcup_{S \in F} S$; $N = |F|$; $Set(i) = S_i$, $1 \leq i \leq N$;
2. If $Un\mathrm{cov} = \varnothing$, algorithm stops and returns $Sub$ as a solution;
3. Select such $j \leq N$ that $|Set(j)|$ is maximal;
4. assume $Sub = Sub \cup \{S_j\}$, $Un\mathrm{cov} = Un\mathrm{cov} - Set(j)$, $Set(i) = Set(i) - Set(j)$, $1 \leq i \leq N$;
5. Return to step 2.

Features of realization: sets $S_i$ are composed from the input matrix, each such set is stored as a list, there is a counter for a number of items for each $S_i$. Function del_el removes the components of set included in the coverage from the sets which are not included in it (delete list items), and decreases the counters. If the counter is zero, the set is excluded from consideration. The program terminates when all counters are zero.

*Climb Algorithm.* Taking into account the results of the structure obtained after a preliminary analysis, we can assume that a solution can be found faster if look through the diagram from the bottom up. It is sufficient to find the first solution on the level and move higher, as all other possible solutions at a level will not be better than the first found one. The algorithm stops when there is no feasible solutions on the level and the solution obtained at the previous level is optimal. This approach is implemented in the program Climb [6].

Features of implementation: at each level, the program consistently generates interchanges (length vector *depth* with camera indices) and simultaneously verifies the inequalities $\sum_{j=1}^{n} a_{ij}x_j \geq b_i$. In order to verify these inequalities, no need to compute the sum, it is enough to compare a row of the matrix and interchange. If the row has at least one unit is in its place determined by the indices in the interchange, the inequality for a given row is valid. When the first feasible solution is found, the check at the current level ends, the resulting solution is stored and there is a transition to a higher level.

The result of the program C1 is useful in Down. The size of the resulting solutions vector, reduced by 1, should be set as an initial parameter $depth$. The program will either find a solution at the levels above, and it will be better, or it will not find, then the resulting vector will be optimal.

*Descend algorithm* [9]. In this problem, it is convenient to look through vertices in a different order in series by the levels in the diagram, i.e. review solutions with the same number of indexes in the list $N_1$. In this case, the very first feasible solution will be one of the best. This method is implemented in the program Descend [10,11].

Implementation is similar to Climb with the difference that the chart levels are in the top. In addition, the program can also use the result of C1: as there is only a slight difference from the accurate solution, the accurate solution vector size must not be less than the integer part of the $1 + \ln(N)$, so exactly this expression should be used as the initial value of *depth*. This algorithm works best on small sparse matrices.

## Calculations and results

Approximation analysis of two districts was done and input matrices were formed for testing (Table 1). Average number of cameras placement points is 20-40. At $N > 40$, only approximation algorithm works effectively, as the accurate algorithm may require complete view of the medial level (where CN = 2> 1010 cells).

The input data for all programs is a Boolean matrix $A(i, j)$ of size $N \otimes M$, formed as follows: $A(i, j) = 1$ when the camera $j$ tracks the area $i$, otherwise $A(i, j) = 0$. The matrix is recorded line by line into the text file, may contain or not gaps between the elements, after entering the matrix, the transition to a new line shall be done.

At construction of the matrix, the columns can be placed in such a manner that the first *k* columns will provide some feasible solution, the program will run faster (because of the peculiarities of interchanges generating). For this purpose, it is sufficient that there is at least one figure of one in each row of the matrix [12].

The matrix shall not contain zero rows and columns. The zero row indicates that a certain area is not visible and the problem has no solution; column zero indicates that some camera does not track any area and therefore it is not needed for coverage.If the matrix contains a row, where a figure of one is found only once, so the area is traced only by one camera and this camera should be included to the coverage obligatorily, therefore, it is possible to remove the row and column from the matrix, at the intersection of which the figure is situated and thus reduce the dimension ( later, this camera shall be added to the coverage). In addition, you need to remove all the other sites viewed by this camera from the sets of areas of other cameras (the rows containing the single elements of the camera shall be deleted).

Programs Descend and Climb compute the size of the input matrix (N and M parameters). The produced result of all programs is an index of the camera included in the coverage.

Calculations were carried out at the processor frequency 1GHz (Table 2).

Table 1 Input matrix of two districts

| Input matrix of district A | Input matrix of district B |
|---|---|
| 1100000000000001000000000000 | 11000000000000000000000000000000 |
| 1100000000000001000000000000 | 01100000000000000000000000000000 |
| 1000000000000001000000000000 | 01000000000000000001000000000000 |
| 1000000000000000000010000100 | 00000000000000000011000000000000 |
| 0100000000000100000000000000 | 00000000000000000010000000000010 |
| 0010000000000100000000000000 | 10000000000000000000000000000010 |
| 0011000000000100000000000000 | 10000000000000000000000000000011 |
| 0001000000001001000000000100 | 00000000000000000110000000000000 |
| 0001000000001001000000000100 | 00000000000000001100000000000000 |
| 0001000000001011000000000100 | 00000000000000011000000000000000 |
| 0001000000000011000000000100 | 00000000000001110000000000000000 |
| 0000100000000100000000000000 | 00000000000001110000000000000000 |
| 0000100000000000000000100000 | 00000000000011000000000000000000 |
| 0000100000000000000001000000 | 00000000011100000000000000000000 |
| 0000010000000000000011000000 | 00000000011000000000000000000000 |
| 1000000000000000000010000100 | 00000000110000000000000000000000 |
| 0000011000000000000110000000 | 00000001100000000000000000000000 |
| 0000010000000001000000000000 | 00000011000000000000000000000000 |
| 0000011000000000000110000000 | 00001100000000000000000000000000 |
| 0001000000000000000100000000 | 00011000000000000000000000000000 |
| 0000011010000000000110000000 | 00110000000000000000000000000000 |
| 0000001000001000000000000000 | 00100000000000000001000000000000 |
| 0010000000000000001000000000 | 00000000000000000001100000000000 |
| 0000001010000000001100000000 | 00000000000000000000110000000000 |
| 0000001010000000001000000000 | 00000000000000000000011000000000 |
| 0000000010000000000000000001 | 00000001000000000000000100000000 |
| 0000001000000000000000000010 | 00000000010000000000000010000000 |
| 0000100000000000000001001000 | 00000000000000000000000011000000 |
| 0000000001000000000000100000 | 00000000000000000000000001100000 |
| 0000000000110000000000000000 | 00000000000000000000000000110000 |
| 0000000000100000000000000001 | 00000000000000000000000000011000 |
| 0000000000110000000000000000 | 00000000000000000000000000001100 |
| 0000000000010000000000000010 | 01000000000000000001000000011000 |
| 0000000000110000000000000000 | 00000000000000000010000000000100 |
| 0000000001000000000000100000 | 00000000000001000000000010000000 |
|  | 00000000000000000100000000110000 |
|  | 00000000000000001100000000000001 |

Table 2. Duration of computing by programs of different algorithms

| Matrix | Program | Work duration |
|---|---|---|
| A | C1 | less than 1 second |
| A | Climb | 1 second |
| A | Descend | 2 seconds |
| A | Climb at depth=10 | 1 second |
| A | Descend at depth=2 | 2 seconds |
| B | C1 | less than 1 second |
| B | Climb | 5 min 52 sec |
| B | Descend | 11 min 23 sec |
| B | Climb at depth=15 | 5 min 31 sec |
| B | Descend at depth=3 | 11 min 23 sec |

## Conclusions

This article describes the measures allowing to divide the existing transport system of the city into parts in a such way that a surveillance system can be implemented in each obtained part by means of the minimum number of cameras and to view all the required areas, using accurate algorithms for the coverage in the calculations.

As a result of the work, the necessary set of measures and programs is obtained, which implements an approximation and adapted accurate algorithms to this prblem. The work can be used for implementation of any kind of video surveillance systems.

***Authors**: prof. dr hab. inż. Waldemar Wójcik, Politechnika Lubelska, Instytut Elektroniki i Technik Informacyjnych, Nadbystrzycka 38A, 20-618 Lublin, E-mail: waldemar.wojcik@pollub.pl ; dr inż.* Orken Mamyrbayev, Institute of Information and Computational Technologies of the Committee of science of MES, Kazakhstan, *E-mail:* morkenj@mail.ru *; mgr inż.* Ainur Akhmediyarova, Kazakh National Research Technical University after K.I.Satpaev, Almaty, Kazakhstan, *E-mail: aat.78@mail.ru; mgr inż.* Dinara Kassymova, Kazakh National Research Technical University after K.I.Satpaev, Almaty, Kazakhstan, *E-mail:* *dika.cat@mail.ru; prof. dr hab. inż.* Irbulat Utepbergenov, Institute of Information and Computational Technologies of the Committee of science of MES, Kazakhstan, *E-mail:* *i.utepbergenov@gmail.com*.

## REFERENCES

[1] Kostomarova V., Foreign experience in the implementation of intelligent transport systems. *Actual problems of the humanitarian and natural sciences*, 4 (2016), No. 4, 110-113.
[2] Kryuchkov V., Tikhomirov A., The development of transport on the basis of the strategy for the implementation of intelligent transport systems. *Transport of the Russian Federation*, 33 (2011) No. 2, 7-10.
[3] Akhmediyarova A., Kassymova D., Utegenova A., Utepbergenov I., Development and research of the algorithm for determining the maximum flow at distribution in the network, *Open Computer Science*, 6 (2016), 208-213
[4] Polyakov K.Yu., The theory of automatic control, *St. Petersburg,* (2008), 4–20
[5] Kristofides N., Graph theory, *"Mir" Publishing house, Moscow*, (1978)
[6] Erzin A., Introduction to Operations Research, *Novosibirsk,* (2006)
[7] Johnson D., Approximation Algorithms for Combinatorial Problems, *Journal of computer and system sciences*, (1974), No. 9, 256-278.
[8] Popkov V., Mathematical model of connectivity, 2nd ed., Rev. and ext., *IMViMG Siberean branch of Russian Academy of Sciences, Novosibirsk*, (2006).
[9] Ageev A., Algorithms with improved accuracy of estimates for the set covering problem, *Discrete Analysis and Operations Research*, 2 (2004), 3-10.
[10] Mashkov V., Smolarz A., Lytvynenko V., The problem of coalition formation modelling, *Informatyka Automatyka Pomiary w Gospodarce i Ochronie Środowiska*, (2014), No. 4, 38-40
[11] Eremeev A, Zaozerskaya L, Kolokolov A., The problem of set covering: complexity - algorithms, experimental studies, *Discrete Analysis and Operations Research*, 7 (2000), No. 2, 22-46.
[12] Mashkov V., Smolarz A., Lytvynenko V., Gromaszek K., The problem of system fault-tolerance, *Informatyka Automatyka Pomiary w Gospodarce i Ochronie Środowiska*, (2014), No. 4, 41-44