GEOMATICS AND ENVIRONMENTAL ENGINEERING • Volume 5 • Number 4 • 2011

Jan Ruchel*, Tomasz Adamczyk*

Application MS Access for Surveying Computations**

1. Introduction

The computations performed in surveying tasks require a proper computer application to be executed. We have a choice here either to buy a commercial program performing needed computations or to build an independent application. We can also use a default database supporting program and combine a system filing the surveying data with a computation module that gives a full capability of individual specifying the details of the computations performed as well as the method and range of presentation of their results.

A properly designed database allows to build a complementary application able to execute any tasks, from the simplest computations to complex analyses. It is us who decides what functions should be realized by the application.

The commercial programs available nowadays at the market present the final results and often do not offer any view into indirect results that can be indispensable data for making an analysis, especially when any computation problems occur. If we build our own application in a form of a database with a computation module we can design it in a way that allows to present the indirect results at any stage of computations.

An application performing the computations we need can be created in many ways. We can design it from a scratch, using existing software-developing tools such as C++, Delphi, Java, Visual Basic (VB) or others. We can also use the popular database management system MS Access. Besides its wide possibilities in managing the surveying database the application includes components that allow to write procedures and combine them into modules with the use of Visual Basic for Applications language (VBA). This gives a possibility of building a complete application for a given group of computation tasks.

* AGH University of Science and Technology, Faculty of Mining, Surveying and Environmental Engineering, Department of Geomatics, Krakow, Poland

** The paper has been worked out within University research program no. 11.11.150.006

Below is presented an example of building a computational application for varied surveying constructions with the use of geodetic database and the VBA language.

The development of hardware influenced on the programming languages development; from low level language (assembler) to the high level ones (mostly C++). At this moment they are less complicated, include many facilities and are more friendly for normal computer users.

The Basic language exists since 1963. First it was a very simple programming language for beginners. A break-through occurred with launching the Visual Basic 1.0. At the same time the Microsoft Windows operational system was introduced. VB became a visual language in which many programming procedures were executed through graphical arrangement of objects on the application window (form). It did not require writing long source codes defining a proper appearance of the application. In VB a programmer could quickly build a fully working application, taking advantage of a graphic user interface. Visual Basic for Applications (VBA) is a programming language based on VB, implemented in the MS Office applications and many others (including AutoCAD). This version of Visual Basic conduces mostly to automation the work with files through macro commands. The main difference between VB and VBA is that the latter does not allow to create independent complex applications of exe type. The code of a program written in VBA is always placed in a file created by a program supporting VBA – for example in a.mdb or.accdb file of the MS Access database. Such a program requires an executable environment i.e. an application handling the given document type to be installed on the programmer's computer.

2. Building an Event-Type Application (Programme Form and Code)

An application realized in VB language consists of two parts:

- graphic user interface, or a system of a window called a form and controls assigned to this window; in easy tasks instead of the form single controls can be used, mostly buttons;
- code of the application, containing procedures with series of instructions to be executed.

Applications built within VB or VBA are event-type programs. That is, a procedure is triggered if an appropriate event occurs. As an event should be regarded user activities such as: clicking a mouse button, pressing a proper key, marking with a cursor etc. Events can be also connected with external devices (for example some measuring apparatus) or with time. Occurrence or non-occurrence of an appropriate event conditions the order of the code execution. That is why a proper order of events is crucial in programming.

If an event is to trigger a wanted reaction, such as executing a line of instructions, it must be connected with a particular control in a form (or in other Access object) of our application.

The procedure of event management has the following syntax:

Private Sub Object_event (optional parameters of the procedure)
...
'list of instructions assigned to the object and event
...
End Sub

As it is shown above, the name of a procedure consists of names of the object and the event assigned. It can also include parameters connected with the object type and event type. Headers of the procedures are generated automatically by VBA.

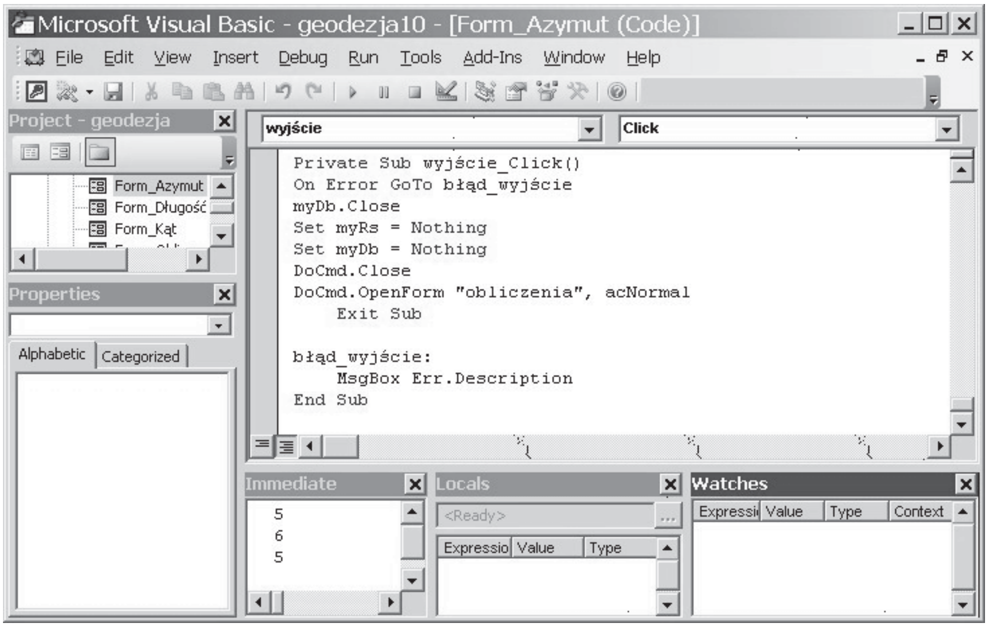
A procedure is a very important element of an application; if any code is to be executed, it must be placed in a procedure. This is the smallest code part that can be launched independently from other code parts.

A module consists of one or more procedures as well as a declaration section, where declarations common for all the module's procedures are put.

A project contains all modules, forms and other objects of the document's parent application and the document itself.

The VBA environment implemented in MS Access is very complex; it consists of many windows allowing to build applications (Fig. 1). The most important windows are:

- Project explorer window – contains all elements of our base and complementary application.
- Properties window – shows current properties of all characteristics for a control activated in the project window. In this window one can modify properties of chosen characteristics for an active control. If needed, the characteristics can be also changed from the code level in time of executing particular procedures.
- Code window – contains created procedures and declarations of our project.
- Windows that support the process of testing and investigating errors (watches, locals and immediate).
- Project of the appearance is realized (not like in VBA within Excel or in VB) as one of the access objects (form type) in which one can design the view of the application (or its parts) in graphical mode (Fig. 2).
- Toolbox window – contains a list of available controls (elementary objects) used in form projecting mode. Also properties window is available for this mode (Fig. 3).

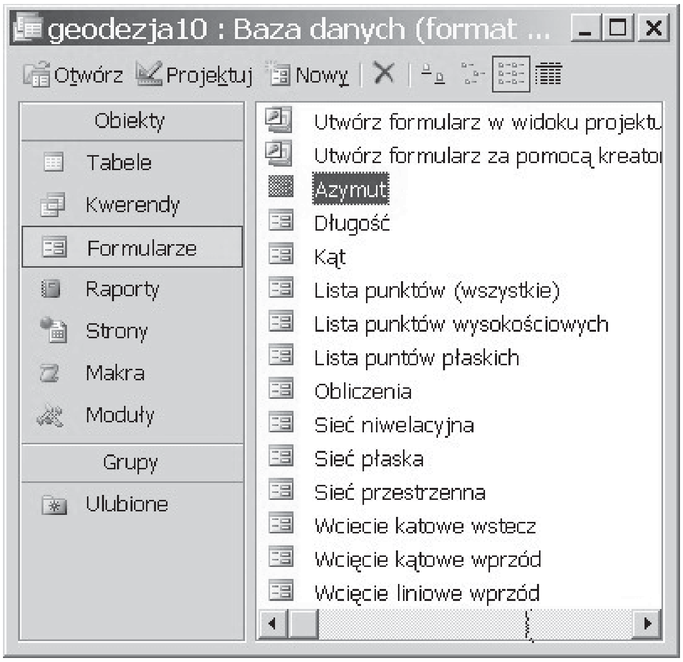

Microsoft Visual Basic - geodezja10 - [Form_Azymut (Code)]
File Edit View Insert Debug Run Tools Add-Ins Window Help
Project - geodezja
Form_Azymut
Form_Długość
Form_Kąt
Properties
Alphabetic
Categorized
wyjście
Click
Immediate
Locals
<Ready>
Expressio Value Type
Watches
Expressi Value Type Context

```
Private Sub wyjście_Click()
On Error GoTo błąd_wyjście
myDb.Close
Set myRs = Nothing
Set myDb = Nothing
DoCmd.Close
DoCmd.OpenForm "obliczenia", acNormal
    Exit Sub

błąd_wyjście:
    MsgBox Err.Description
End Sub
```

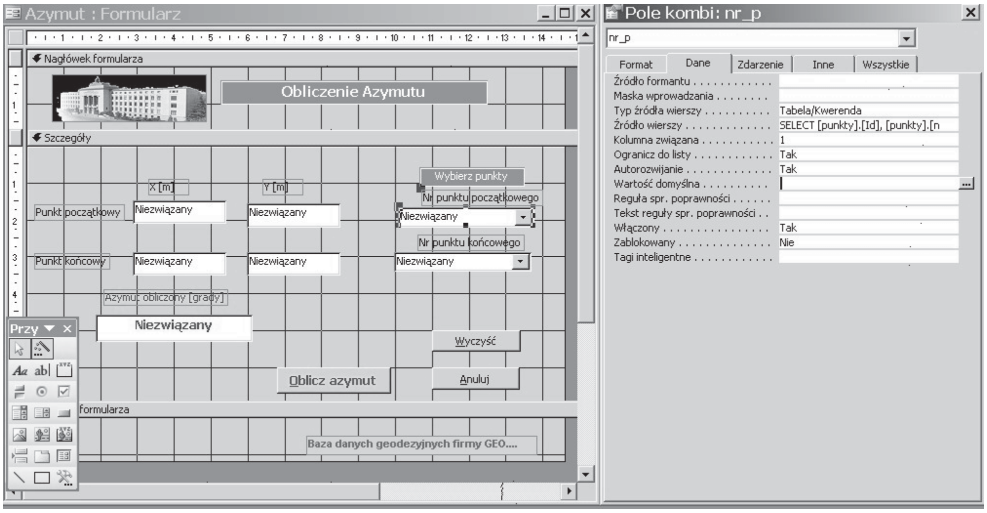
```
5
6
5
```

Fig. 1. The environment of code execution for a complementary application in VBA



Utwórz formularz w widoku projektu
Utwórz formularz za pomocą kreatora
Azymut
Długość
Kąt
Lista punktów (wszystkie)
Lista punktów wysokościowych
Lista puntów płaskich
Obliczenia
Sieć niwelacyjna
Sieć płaska
Sieć przestrzenna
Wciecie katowe wstecz
Wcięcie kątowe wprzód
Wcięcie liniowe wprzód

Fig. 2. Database window for objects of form type, where creating and modifying forms for database and complementary application is possible



Źródło formantu	
Maska wprowadzania	
Typ źródła wierszy	Tabela/Kwerenda
Źródło wierszy	SELECT [punkty].[Id], [punkty].[n
Kolumna związana	1
Ogranicz do listy	Tak
Autorozwijanie	Tak
Wartość domyślna	
Reguła spr. poprawności	
Tekst reguły spr. poprawności	
Włączony	Tak
Zablokowany	Nie
Tagi inteligentne	

Fig. 3. Window for projecting form for a complementary application, together with toolbox and properties windows

3. Creating Event Procedures

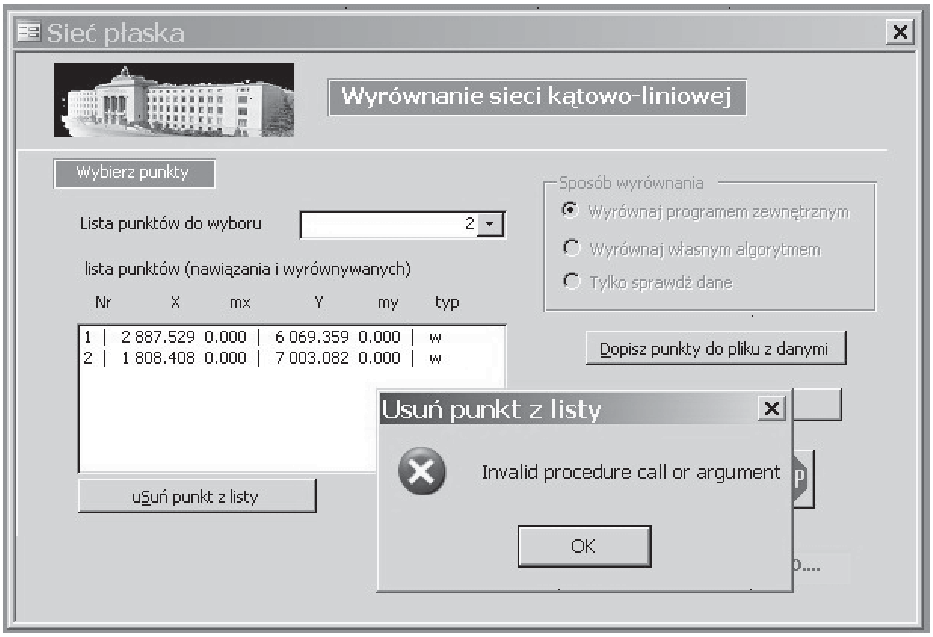
Technique for writing event procedures is quite simple and requires medium programming level. To create an event procedure for chosen control from a form one has to follow these steps:

1. Launch Visual Basic Editor.
2. In the project window choose an appropriate form for which the event procedure is to be created. Code window for the chosen spreadsheet will appear.
3. In code window, from object drop-down list choose a control for which a procedure will be created. On choosing a control MS Access automatically creates a spreadsheet of a default event procedure. For most of controls a default event is click (clicking a mouse button).
4. Other than default event can be chosen from event drop-down list (right top of code window). It contains all possible events for a control chosen. On choosing a particular event an appropriate spreadsheet of an event procedure is created.

To add a control to a form one has to pick it from toolbox window with a mouse cursor, place it in a form and define its size and position using mouse and graphical mechanisms of Windows environment. In properties window one can change properties of particular characteristics for a chosen control.

VBA has an error control inbuilt. Errors can be of several types:

- Editing text errors can be omitted by thorough checking of the text written. These errors are not seriously bad for an application's work.
- Compilation errors make compilation of an application impossible. They come from improper syntax of declarations or instructions, or mistaken identification of particular elements of the program.
- Execution errors appear during executing an algorithm. To prevent them specific securities (error traps) should be used during writing the code. Message box with error type signalization is used very often (Fig. 4).
- Logic errors are the most dangerous, for they cause incorrect computation results while the program seems to be working properly. They are also the most difficult ones to detect, for there is no messages that would report their occurrence. Improperly constructed algorithms are usually the cause of logic errors.


Sieć płaska
Wyrównanie sieci kątowo-liniowej
Wybierz punkty
Dopisz punkty do pliku z danymi
uSuń punkt z listy
OK

Lista punktów do wyboru: 2

lista punktów (nawiązania i wyrównywanych)

```
Nr     X        mx    |    Y        my    |  typ
1 |  2 887.529  0.000 |  6 069.359  0.000 |  w
2 |  1 808.408  0.000 |  7 003.082  0.000 |  w
```

Sposób wyrównania
- [✓ Wyrównaj programem zewnętrznym]
- Wyrównaj własnym algorytmem
- Tylko sprawdź dane

Usuń punkt z listy

Invalid procedure call or argument

Fig. 4. Error signalization window – invoking code (after fulfilling proper conditions)

Unfortunately, compilation process support and application testing in VBA implemented into MS Access is vaguely limited compared to the VBA version within MS Excel or to the VB.

When undertaking to build an application, one should have a very thoroughly defined expectations concerning the aims of the application, as well as prepared and tested algorithm of the proceedings to gain the aim. To prepare the application to compute one has to design proper structures, within which the data will be stored (and also reserve proper memory resources for the data).

4. An Exemplary Application Designed to Execute Surveying Computations (Including Adjustment the Surveying Networks) with the Use of Database

Every person acknowledged with basics of programming is capable of building, with the help of VBA, an application that can execute any surveying computation processes. Of course, to build such an application, the knowledge concerning the task itself is indispensable; one must know the algorithm solving the problem in case. The illustration given in this paper is the application complementary to the Geodetic Database, called "Geodetic computations" ("Obliczenia geodezyjne"). It allows to execute basic surveying computations and to adjust networks, including horizontal networks according to the least square method of adjustment algorithm.

The application gives access to various elements of the computations. The main rule is a free choice of points (proper coordinates) from a database resources, or possibility of entering coordinates independent from the database resources (Fig. 5). This regards also other elements, for example observations.

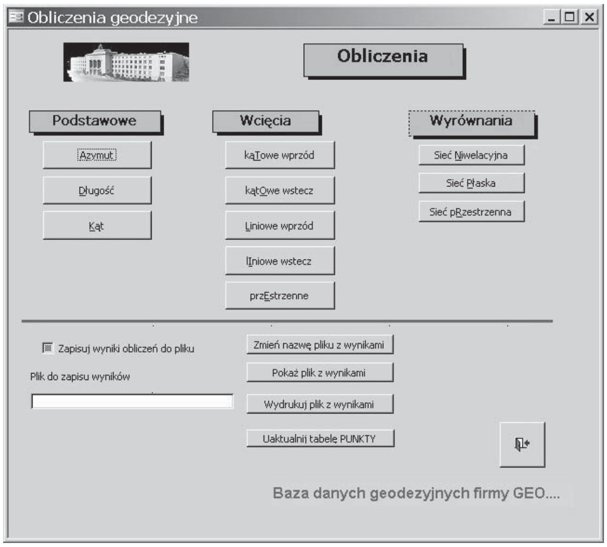
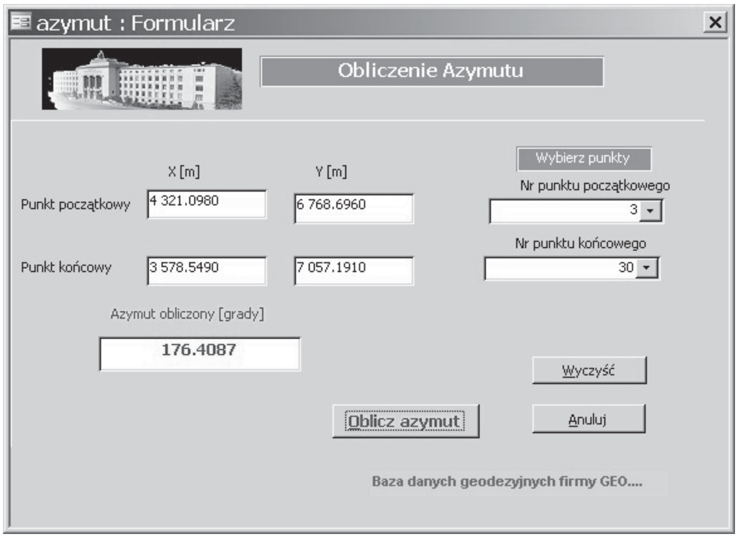


Fig. 5. Application's main window – it lists types of possible computations

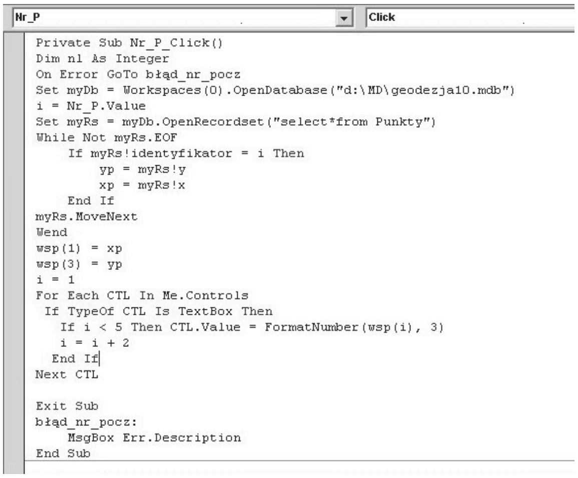
An example of basic computations can be computing an azimuth. The algorithm in this case is obvious and does not need a presentation. Computations on the azimuth form are presented on figures 6 and 7 shows the algorithm of the procedure that chooses a proper point from database resources (from the "Points" ["Punkty"] table, which contains coordinates). The choice is made by indicating a point from a drop-down list. Indicated point is automatically loaded into according cells on the form.



	X [m]	Y [m]	
Punkt początkowy	4 321.0980	6 768.6960	Nr punktu początkowego: 3
Punkt końcowy	3 578.5490	7 057.1910	Nr punktu końcowego: 30

Azymut obliczony [grady]: 176.4087

Fig. 6. Computing azimuth for coordinates chosen from the list



```
Private Sub Nr_P_Click()
Dim nl As Integer
On Error GoTo błąd_nr_pocz
Set myDb = Workspaces(0).OpenDatabase("d:\MD\geodezja10.mdb")
i = Nr_P.Value
Set myRs = myDb.OpenRecordset("select*from Punkty")
While Not myRs.EOF
    If myRs!identyfikator = i Then
        yp = myRs!y
        xp = myRs!x
    End If
myRs.MoveNext
Wend
wsp(1) = xp
wsp(3) = yp
i = 1
For Each CTL In Me.Controls
 If TypeOf CTL Is TextBox Then
   If i < 5 Then CTL.Value = FormatNumber(wsp(i), 3)
   i = i + 2
  End If
Next CTL

Exit Sub
błąd_nr_pocz:
    MsgBox Err.Description
End Sub
```

Fig. 7. The code window containing the procedure for choosing a given point from the list (as a starting point for the azimuth calculation)

More complex forms and, of course, procedure codes, are needed for the network adjustment. Our example will be the surveying network adjustment computation.

The basic formulas used in this program are listed below:

– Observation weights:

$$p_i = \frac{1}{m^2}.$$

– The system of correction equations is generated using the following relation:

• for linear observations:

$$w_i = dD_i + D_{approx.} - D_{measured},$$

where:

$$dD_i = \begin{vmatrix} dx_P & dy_P \\ -\cos A_{P-K} & -\sin A_{P-K} \end{vmatrix}\begin{vmatrix} dx_K & dy_K \\ \cos A_{P-K} & \sin A_{P-K} \end{vmatrix},$$

P – starting point,
K – ending point,
D_i – observation measured,
w_i – distance correction,
$D_i^0 = 'PK' = \sqrt{\Delta x^2 + \Delta y^2}$ – approximated distance,
A_{P-K} – azimuth for the side measured,
dx_P, dx_K, dy_P, dy_K – partial coordinates;

• for angular observations:

$$v_i = d\alpha_i + \alpha_{approx} - \alpha_{measured},$$

where:

$$d\alpha_i = \begin{vmatrix} dx_L & dy_L \\ A_L & B_L \end{vmatrix}\begin{vmatrix} dx_P & dy_P \\ -A_P & -B_P \end{vmatrix}\begin{vmatrix} dc_C & dy_C \\ -(A_L - A_p) & -(B_L - B_P) \end{vmatrix},$$

α_i – observation measured,
v_i – angle correction,
α_i^0 – approximated angle,

A_L, B_L, A_P, B_P – gradients respectively for the left (L) and right (P) arm of the angle:

$$A = \frac{\Delta x}{\Delta x^2 + \Delta y^2} \cdot \rho, \quad B = \frac{\Delta y}{\Delta x^2 + \Delta y^2} \cdot \rho,$$

$dx_L, dx_C, dx_P,$
dy_L, dy_C, dy_P – partial coordinates.

– The correction equations in the matrix form:

$$V = A \cdot x + w.$$

– The solution of the system of standard equations:

$$(A^T \cdot p \cdot A) \cdot X + (A^T \cdot p \cdot w) = 0,$$

$$-x = (A^T \cdot p \cdot A)^{-1} \cdot (A^T \cdot p \cdot w).$$

– Computation of adjusted coordinates:

$$X_i = X_i^0 + dx_i,$$

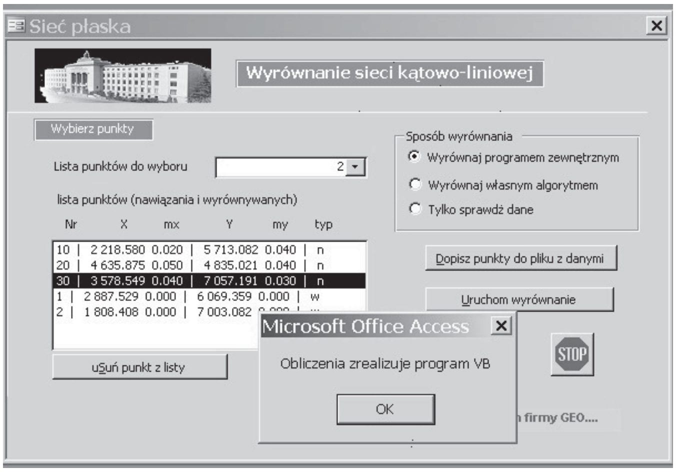
$$Y_i = Y_i^0 + dy_i.$$

– Basic analysis of accuracy:

$$m_0 = \pm \sqrt{\frac{[pvv]}{n-u}},$$

$$\text{cov}(x) = m_0^2 \cdot (A^T \cdot p \cdot A)^{-1}.$$

For the algorithm presented above there is created a code that allows to adjust a network. The results can be displayed on a computer monitor in a window. The application gives options of saving the results as text files in a chosen format and with freely indicated range. The adjustment itself can be executed with an appropriate module of a complementary application (internal) or with an external tool (for example an adjusting application built with VB). The way of executing the adjustment is chosen by indicating a proper option from the list at the start of computations. An exemplary window with an application thus realized is shown on figure 8.



Lista punktów do wyboru: 2

lista punktów (nawiązania i wyrównywanych)

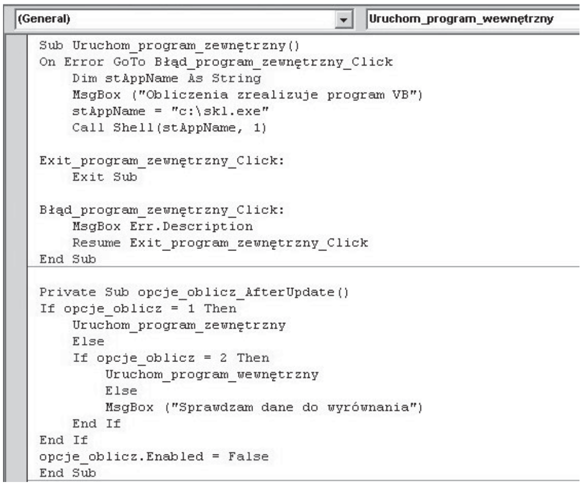
```
 Nr       X       mx          Y       my    typ
10 |  2 218.580  0.020 |  5 713.082  0.040 |  n
20 |  4 635.875  0.050 |  4 835.021  0.040 |  n
30 |  3 578.549  0.040 |  7 057.191  0.030 |  n
1  |  2 887.529  0.000 |  6 069.359  0.000 |  w
2  |  1 808.408  0.000 |  7 003.082
```

Sposób wyrównania
- [✓ Wyrównaj programem zewnętrznym]
- Wyrównaj własnym algorytmem
- Tylko sprawdź dane

Obliczenia zrealizuje program VB

Fig. 8. The surveying network adjustment window – an external application has been chosen to execute the adjustment

The module that executes the surveying network adjustment consists of properly organized procedures and functions, mutually connected with regard to various relations, thus making the adjustment calculation with appropriate checks possible. Figure 9 shows a fragment of the module, containing a code of a procedure of adjustment options choice and a procedure of external application choice. The adjustment results are displayed on a monitor and saved into text files (Fig. 10) for a presumptive further use. The coordinates of adjusted points can be added to the database resource; as new points or as actualized values of coordinates and errors of points already stored.



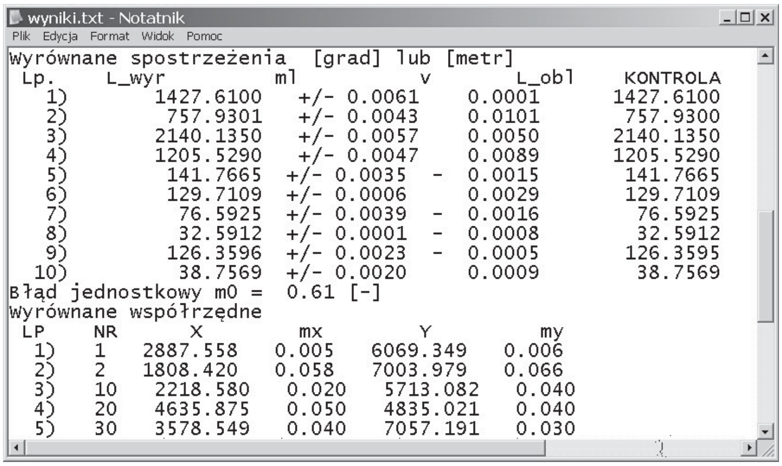
```
Sub Uruchom_program_zewnętrzny()
On Error GoTo Błąd_program_zewnętrzny_Click
    Dim stAppName As String
    MsgBox ("Obliczenia zrealizuje program VB")
    stAppName = "c:\skl.exe"
    Call Shell(stAppName, 1)

Exit_program_zewnętrzny_Click:
    Exit Sub

Błąd_program_zewnętrzny_Click:
    MsgBox Err.Description
    Resume Exit_program_zewnętrzny_Click
End Sub

Private Sub opcje_oblicz_AfterUpdate()
If opcje_oblicz = 1 Then
    Uruchom_program_zewnętrzny
    Else
    If opcje_oblicz = 2 Then
        Uruchom_program_wewnętrzny
        Else
        MsgBox ("Sprawdzam dane do wyrównania")
    End If
End If
opcje_oblicz.Enabled = False
End Sub
```

Fig. 9. Window with the code executing the surveying network adjustment (a fragment)



```
Wyrównane spostrzeżenia  [grad] lub [metr]
 Lp.    L_wyr            ml           v          L_obl      KONTROLA
   1)       1427.6100   +/- 0.0061    0.0001      1427.6100
   2)        757.9301   +/- 0.0043    0.0101       757.9300
   3)       2140.1350   +/- 0.0057    0.0050      2140.1350
   4)       1205.5290   +/- 0.0047    0.0089      1205.5290
   5)        141.7665  +/- 0.0035  -  0.0015       141.7665
   6)        129.7109  +/- 0.0006     0.0029       129.7109
   7)         76.5925  +/- 0.0039  -  0.0016        76.5925
   8)         32.5912  +/- 0.0001  -  0.0008        32.5912
   9)        126.3596  +/- 0.0023  -  0.0005       126.3595
  10)         38.7569  +/- 0.0020     0.0009        38.7569
Błąd jednostkowy m0 =  0.61 [-]
Wyrównane współrzędne
 LP     NR      X         mx       Y          my
  1)    1    2887.558   0.005   6069.349   0.006
  2)    2    1808.420   0.058   7003.979   0.066
  3)    10     2218.580   0.020    5713.082   0.040
  4)    20     4635.875   0.050    4835.021   0.040
  5)    30     3578.549   0.040    7057.191   0.030
```

Fig. 10. Text file with results of network adjustment computation (a fragment)

5. Summary

Using Visual Basic for Applications language for building applications gives many advantages. Building with the help of MS Access allows to combine two functions: building and managing a database with field data and measurement results recorded in any system of tables, and executing various computations, including network adjustments. The project can be extended (enhanced with additional functions) with no limits. It allows to view indirect and final results in any chosen form. It also allows to add the results to the database resources.

We bear no additional costs except for our labour time (assuming we own MS Office suite).

Some drawback, in comparison to building an application with the use of VB, is a bit modest error support in the VBA version implemented into MS Access.

It is recommended to use test files in the process of computations. These files, created during the process, are of high utility. They are very suitable and friendly to be viewed and edited and can be also used in other applications. Especially easy and suitable can be cooperation with AutoCAD, which is also equipped with a VBA implementation.

References

[1] Bluttman K., Freeze W.: *Access. Analiza danych. Receptury*. Helion, Gliwice 2008.

[2] Czaja J.: *Modele statystyczne w informacji o terenie*. Wydawnictwa AGH, Kraków 1996.

[3] McFedries P.: *Access 2007 PL. Formuły, raporty, kwerendy.* Helion, Gliwice 2009.

[4] Hausbrandt S.: *Rachunek wyrównawczy i obliczenia geodezyjne. Tom 1–2*. PPWK, Warszawa 1971.

[5] Jasicska E., Ruchel J.: *Using a Spreadsheet for Surveying Computations*. Geomatics and Environmental Engineering, vol. 4, no. 4, pp. 77–87.

[6] Tor A.: *Access 2003. Tworzenie aplikacji*. Studio Komputerowe Tortech, Warszawa 2008.

[7] Tor A.: *Access 2003. Visual Basic*. Studio Komputerowe Tortech, Warszawa 2009.

[8] Willet E.C., Cummings S.: *ABC Visual Basic dla aplikacji w Office XP*. Helion, Gliwice 2002.