DIPLOMAMUNKA

Robot mérőállomásokra alapozott mozgásvizsgálati rendszer továbbfejlesztése és gyakorlati alkalmazása

Moka Dániel

Budapesti Műszaki és Gazdaságtudományi Egyetem
Építőmérnöki Kar, Általános- és Felsőgeodézia Tanszék
Építőipari geodéziai szakirány
2012

Köszönetnyilvánítás

Nagyon sok köszönettel tartozom Dr. Siki Zoltán tanár úrnak, aki lehetővé tette számomra, hogy betekintést nyerjek az általa kialakított projektbe, melynek használatával a szakdolgozatom elkészítése lehető vált. Köszönöm tanár úrnak a félév során nyújtott segítő munkáját, odaadását és támogatását, valamint esetenként a hosszabb konzultációkat és találkozókat, melyek nagymértékben hozzájárultak a feladat sikeres végrehajtásához.

Nyilatkozat:

Alulírott Moka Dániel a Budapesti Műszaki és Gazdaságtudományi Egyetem hallgatója kijelentem, hogy a diplomamunkát meg nem engedett segítség nélkül, saját magam készítettem, és a dolgozatban csak az irodalomjegyzékben megadott forrásokat használtam fel. Minden olyan részt, melyet szó szerint, vagy azonos értelemben, de átfogalmazva más forrásból átvettem, egyértelműen,a forrás megadásával megjelöltem.

Budapest, 2012. december 8.

……..

Moka Dániel

Abstract

Developing an Automatic Deformation Monitoring System Based on Robotic Total Stations within Practical Applications

We live in a dynamic world. Buildings, bridges, dams, and every facilities are in permanent movement, that we must detect, in order to preserve the objects from incidental and occuring damages. On the other hand we greatly want to prevent the loss of large amount of money and last but not least we have to care about people's life who are around the construction. Engineers more often and repeatedly use Robotic Totals Stations (RTS) to solve this problem, nowadays. As a matter of fact, RTS is not enough alone, we must build up an Automatic Deformation Monitoring System (ADMS) using RTSs.

The first aim of my diploma work is to present operations of AMDSs, giving some examples of these systems from the world. After that I would like to introduce the Ulyxes project, which is developed by Dr. Siki Zoltán from Department of Geodesy and Surveying in Technical University of Budapest. Ulyxes is an open source projet to drive RTSs and publish the results of the measurements on the internet. During my work, i have used and developed the so-called „TclAPI" which is core part of Ulyxes system. Applying this, we are able to control and manage robotic total stations therefore we can perfrom automated measurements on different objects. At the end of my diploma, there are four practical applications, demonstrating the efficiency of TclAPI and robotic total stations.

Tartalomjegyzék

1. Bevezetés

Manapság a robot mérőállomásokat egyre szélesebb körben alkalmazzák a mérnökgeodéziában. Használatuk nagymértékben megkönnyítik a mérnökök munkáját, valamint a feladat megoldására szánt időt is szignifikánsan lerövidítik. Ezen robot mérőműszerek alkalmazása számos új, izgalmas lehetőséget kínálnak a felhasználónak, ilyen például egy munkagép irányítása a terepen egy megadott terepmodell alapján, vagy egy adott létesítmény deformációjának az automatizált mérése. A robot mérőműszer önmagában nem elég, kell hozzá egy megfelelő mozgásvizsgálati rendszert létesíteni, melynek segítségével kezelni lehet a mérés menetét, végrehajtását, az adatok rögzítését és végül, de nem utolsó sorban adatok feldolgozását is. Az ilyen mozgásvizsgálati rendszerek hatékonyságának a bizonyítására számos hazai és külföldi példa szolgál, szerte a világon. A vagyon, ember és élet megóvásának a szerepe a robot mérőállomásoknak, és az ezekkel szorosan összekapcsolódó mozgásvizsgálati rendszereknek köszönhetően a mindennapi életben sokkal nagyobb mértékben biztosított. Erre kitűnő példa a 4-es metró építése során alkalmazott mozgásvizsgálati rendszer alkalmazása, ahol az erre illetékes személyek SMS-ben értesítés kapnak akkor, ha egy adott mozgásvizsgálati pont mozgása egy megadott határértéket túllép.

A szakdolgozatom szöveges része 4 nagy részből tevődik össze, melyek szorosan összekapcsolódnak. Az 2. fejezetben a robot mérőállomások által nyújtotta technológiai lehetőségeket ismertetem az mérnökgeodézia feladatokban, rövid történelmi áttekintéssel. Ebben a fejezetben fogok kitérni a különböző technológiákra, melyek segítségével képesek lehetünk a robot mérőállomásokat programozni és ezen keresztül automatizálni illetve vezérelni a működésüket. A 3. fejezetben az általános mozgásvizsgálati feladatok jellemzését követően a robot mérőállomások által kialakítható mozgásvizsgálati rendszereket és összetételüket fogom bemutatni mind hardver és mind szoftver komponensek terén. Természetesen világszerte több ismert gyártó és cég is foglalkozik az automatizált rendszerek kiépítésével és fejlesztésével, azonban a szakdolgozatom során két, a területen kiemelkedő cégnek, a SolData és a Leica Geosystems rendszereit és szabadalmait fogom ismertetni. Majd ezt követően a fejezet végén lehetőséget nyújtok az olvasónak, hogy betekintsen néhány hazai valamint külföldi automatizált monitoring rendszer funkcionalitásába és működésébe, melyeket az előbb említett cégek hoztak létre.

A 4. fejezetben ismertetem a gyakorlati alkalmazásaim során felhasznált műszerek és eszközök vizsgálatát és kalibrálását, melyeket a munkálataim során az 5. fejezetben lévő **Gyakorlati**

alkalmazások című fejezetben használtam fel. És végül az utolsó fejezetben ismertetni fogom szakdolgozatom fő részét képező, a Budapesti Műszaki Egyetem Általános és Felsőgeodézia tanszéken oktató Dr. Siki Zoltán tanár úr által kialakított Ulyxes névre hallgató projektet, mely egy valódi mozgásvizsgálati rendszer kialakítását foglalja magában. Ennek a rendszernek a szerves részét képező TclAPI (Tcl-Application Programming Interface) kiváló lehetőséget nyújtott számomra, hogy megismerkedjek a robot mérőállomások programozásával és a velük való mérések automatizálásával, valamint ehhez kapcsolódó feladatokkal illetve fejlesztési ötletekkel. Az API használatával elérhető lehetőségeket a fejezet végén, 4 gyakorlati feladat megoldásának keretén belül fogom bemutatni, összevetve más geodéziai technikákkal.

A szakdolgozatom írása és készítése során felhasznált szoftverek nyílt forráskódú eszközök, melyek az internetről szabadon letölthetőek és használhatók.

2. Robot mérőállomások által nyújtotta technológiai lehetőségek a mérnökgeodéziában

2.1. Történeti áttekintés

Az elmúlt fél évszázadban a földmérők által használt teodolitok nagy fejlődéseken mentek keresztül. Az első innovációs műszerek, az elektronikus teodolitok az 1970-es években jelentek meg, melyek felváltották a hagyományos optikai műszereket. Korábban a teodolit és a távmérő külön műszer volt, és az adatrögzítés még manuálisan történt, de később az 1980-as években egy új termék jelent meg a piacon, a mérőállomás (Total Station). Ezen műszereket már olyan elektronikus távolságmérővel (Electronic Distance Meter) szerelték fel, mely a távolságméréshez általában még pontosabb eredményt szolgált. Ezt követően 1990-ben a svédek bemutatták a robot mérőállomásokat (Robotic Total Station) tartalmazó Geodimeter műszercsaládot, mely eszközöket olyan funkciókkal láttak el, mint például az automata célfelismerés és célkövetés. Ezt követően pedig a robot mérőállomások gyártása a mai napig nem állt még le, évről évre újabbnál újabb eszközök jelennek meg a piacon, melyek használhatósága egyre jobb a mérnökgeodéziai munkálatokban

2-1. ábra Geodimeter [AUSS]

2.2. Robot mérőállomások jellemzése

Napjainkban a robot mérőállomást nagy magabiztossággal nevezhetjük a legsokoldalúbb konstrukciónak a piacon. Alkalmazásuk nagymértékben elősegítik a földmérők és az egyes cégek munkáját valamint előszeretettel alkalmazzák kis elmozdulások és alakváltozások mérésére a mérnökgeodéziai mozgásvizsgálatokban.

2-2. ábra Sokkia, Leica, Trimble és Topcon robot mérőállomások [TARP]

„A robot mérőállomásosok előre beprogramozott szög- és iránymérés, távmérés és adatrögzítés végrehajtására képes elektronikus műszerek.”[TARP] Az állótengely körül elforgatható alhidádé és az ebbe beépített, a fekvőtengely körül elforgatható geodéziai távcső szervomotorok segítségével forgatható. A robot mérőállomások megjelenését követően új fogalmak jelentek meg a földméréstan szakszótárában, melyek szorosan összekapcsolódnak az egy-emberes geodézia feladatok végrehajtásához.

Automatikus célfelismerés (Automatic Target Recognition-ATR): Az ATR jelentősége abban rejlik, hogy a felhasználónak nem szükségszerű pontosan megirányozni egy adott prizmát, mivel a finom irányzást az műszer magától elvégzi. Az automatikus célfelismerés tehát lehetővé teszi, hogy a prizmákat minden egyes adandó alkalommal szabatosan megirányozzuk, így ennek köszönhetően nagyon kis elmozdulásokat is észlelni tudunk. Az ATR az automatizált mozgásvizsgálati rendszereknek a működéséhez nagymértékben hozzá járul. Előnyei:

- állandó precíz irányzás a felhasználótól függetlenül
- az irányzás gyors, észlelő fáradságától független végrehajtása
- nem szükséges a parallaxis teljes megszüntetése
- bármilyen prizmával kompatibilis a rendszer.

Automatikus célkövetés (LOCK mode): Az automatikus célkövetésnek köszönhetően, a prizma helyzetét leíró adatok bármely pillanatban meghatározhatók anélkül, hogy a célkövetés megszakadna. Az egyetlen feltétele, hogy az első mérést a prizmára manuálisan kell elvégezni. Használatából adódó kényelmesség és szabadság 360°-os prizma alkalmazásával érhető el a leghatékonyabban, mivel ebben az esetben a prizmát nem kell közvetlenül a műszer felé fordítanunk. A 360°-os prizma egy olyan mérőfelszerelés, amely hat darab összeépített háromszögletű prizmát tartalmaz, melyek lehetővé teszik a megfelelő mérési pontosságot, bármely oldalról. A LOCK mode alkalmazása a robot mérőállomással történő munkagép-vezérlésnél központi szerepet játszik. Felhasználási területe:

2-3. ábra Leica 360°-os prizma [LEIC]

- topográfiai felmérések
- kódolt felmérések további térinformatikai alkalmazásokhoz
- kitűzések elvégzése

Távirányító rendszer (Remote Control System): A kezelő távirányító segítségével képesek vagyunk a mérést távolról is elvégezni. A kezelő és a robot mérőállomás közti kapcsolatot rádiós modem vagy Bluetooth segítségével alakítatjuk ki. Az egyemberes geodéziai munkálatokat gyakorlatilag a távirányító rendszer teszi teljessé, használatával a felméréseinket, kitűzéseinket tehetjük gyorsabbá valamint a GPS-vevő vezérlését is végrehajthatjuk vele. [TARP]

2-4. ábra Robot mérőállomást vezérlő távirányító [CONS]

Image Station: Az Image Station a robot mérőállomás és az fényképezőgép egyesítéséből jön létre. Az Image Station számos új lehetőséget biztosít a földmérő számára. Elsőként is a használatával a felhasználó képes az érintőképernyőn keresztül történő irányzásra. Az irányzás elvégzéséhez elegendő a kijelző képén megjelenő kamerakép bármely részét kijelölni, majd a szervo motor segítségével a műszer a helyes irányba fordul. Bár ennek a pontossága nem közelíti meg a távcső segítségével történő irányzás pontosságát, azonban durva irányzások végrehajtására tökéletesen megfelel. Másfelől pedig akár 360°-os tartományban képes fényképet készíteni, ezzel kép információt is szolgáltatva egy adott munkához. A fényképek készítéséhez, műszergyártóktól függően az egyes robot mérőállomások akár 20x-os nagyítással is rendelkezhetnek, ezzel részlet gazdag képet szolgáltatva a felhasználó számára.

Annak az érdekében, hogy a robot mérőállomásokat vezérelni, irányítani tudjuk illetve automatizált méréseket tudjunk végrehajtani vele, szükség van a műszerek programozására, melynek a részletezését a következő fejezet taglalja

2.3. Robot mérőállomások programozása

A 21. században az építőmérnöki tevékenység igen sokrétűvé vált, sokkal összetettebb és bonyolultabb szerkezetek létesítenek nap, mint nap, melyeknek tervezése, építése és vizsgálata komolyabb feladat elé állítja szakembereket. Az ilyen komplikált szerkezetek megkövetelik, hogy a hagyományos technológiák helyett, modernebb, esetenként újabb módszereket alkalmazzanak az egyes építésirányítási és mozgásvizsgálati feladatokhoz. Az ilyen újszerű módszerek közé tartozik a robot mérőállomások programozása, amely napjainkban az egyik legfejlettebb technológiai „eszköz”, és amelynek köszönhetően, az ehhez értő építőmérnököknek a szakmai lehetőségei szinte korlátlan.

A különböző robot mérőeszközök programozására különféle igények kielégítése, valamint az ezzel a technológiával elérhető előnyök kihasználása miatt lehet szükség. Elsőként is, egyre több létesítmény, szerkezet deformáció mérésénél illetve mozgásvizsgálatánál van igény a folyamatos észlelésekre. Valójában csak egy megfelelően programozott mérőállomásokat tartalmazó mozgásvizsgálati rendszer képes ezt az igényt kielégíteni. Másfelől pedig esetenként szükség lehet a mérési eredmények azonnali kiértékelésére és továbbítására is, az egyes létesítmények megrongálódásának valamint az emberi élet megvédésének érdekében. A robot mérőállomások programozásának egy másik fontos előnye, hogy a mérések még rossz látási viszonyok (pl. éjszaka) között is automatizálhatók. Vannak olyan esetek is, melyeknél a mérési körülmények veszélyes vagy egészségre károsak lehetnek az észlelőre, ilyen lehet például egy atomerőmű közvetlen közelébe való folyamatos mérés, ekkor a robot mérőállomások használata célszerű.

Az egyes mérőállomások automatizál vezérlésére három lehetőség adott, melyek a következők:

1. Mérőállomásra gyárilag telepített szoftverrel (pl. Trimble 5503 Field Applications, Robotic Lite, Leica TCA monitoring): Ennek a megoldásnak az előnye abban rejlik, hogy nincs szükség további programozásra. Hátránya viszont az, hogy az adott műszergyártó határozza meg az alkalmazható lehetőségeket, valamint hogy az adatokat csak utólag olvashatjuk ki a műszerből, ezzel elveszítve azt a lehetőséget, hogy az eredményeket már a helyszínen feldolgozzuk, kiértékeljük.
2. Mérőállomásra feltölthető egyedi fejlesztésű szoftver: Ebben az esetben feltölthető szoftverek már rendelkeznek azzal a tulajdonsággal, hogy az egyéni igények kielégíthetők legyenek, azonban az adatok feldolgozása nem oldható meg teljes körűen, mivel a fedélzeti számítógép kapacitása korlátozott. Sajnos ezért a mért adatokból

készíthető grafikonok rajzolására és különböző, összetett számítások elvégzésére már nincs mód.

3. Számítógépről vezérelt mérés: Ennél a lehetőségnél a számítógépen (laptopon) futó program vezérli a műszer működését. Az egyes eszközökhöz gyakran készítenek alkalmazás-programozói felületet (Application Programming Interface - API), mely egyszerűbbé teszi a programozott műszervezérlést a számítógépen keresztül. Az ehhez hasonló számítógépes programok általában egyedi fejlesztésűek, melyeknek a legfőbb előnye, hogy csak a képzeletünk szab határt a program lehetőségeinek. Mi választhatjuk meg a számítógépes környezetet, az operációs rendszert (Operation System - OS), melyen dolgozni szeretnénk a mérések során, ezzel kihasználva az egyes OS-ek előnyeit, valamint a programnyelvet is mi határozhatjuk meg. További előnyt jelent az is, hogy a mérőműszerrel mért adatok rögtön átkerülhetnek a felhasználó számítógépére, ahol további összetett számításokat végezhetünk rajtuk. A szakdolgozatom során, főként a gyakorlati alkalmazások elvégzéséhez, egy konkrét API-t használtam, melynek az egyes lehetőségeit és tulajdonságait később részletezni is fogok.

Az egyes mérőeszközök programozása még nem elegendő ahhoz, hogy az adott műszert sikeresen alkalmazzuk a mérnökgeodéziai tevékenységeink során, különféle feltételeknek kell teljesülnie ahhoz, hogy egy adott feladat teljesen kivitelezhető és jól megoldható legyen. A feltételek a következők lehetnek:

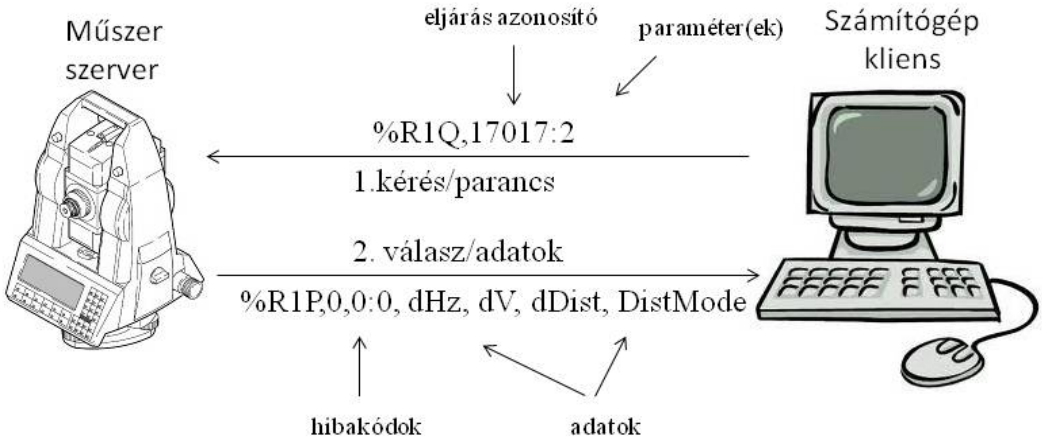
1. Megfelelően telepített, és tájékozott mérőállomás;
2. Szervo motorral rendelkező, automatikus irányzásra és követésre képes robot mérőállomások, melyek lehetővé teszik a műszerek automatizált mozgatását;
3. Kommunikációs csatorna és protokoll a számítógép és a műszer közötti adatcsere lebonyolítására (RS-232, BlueTooth, USB);
4. Vezérlő szoftver:
    a. gyárilag telepített;
    b. egyedi fejlesztésű;
    c. API illetve ezt használó egyedi fejlesztésű program megléte;
5. Szabatos mérést lehetővé tevő, telepített prizmák;
6. Monitoring vizsgálatok esetén szükség lehet meteorológiai szenzorokra, melyeknek köszönhetően az meteorológiai adatok (hőmérséklet, légnyomás, parciális páranyomás) meghatározhatók az egyes távolságméréssel kapcsolatos javításokhoz;

7. Célprizma nélkül történő automatizált felmérések (pl.: földtömegszámítás) elvégzéséhez nélkülözhetetlen a prizma nélküli távmérő a műszerben.

A digitális adatok továbbításának a robot mérőállomások felé több módja létezik. Az adatok megfelelő sebességgel történő átviteléhez a leghasználatosabb módszerek a soros vonal, USB és a BlueTooth. A szakdolgozatom gyakorlati alkalmazásainál a műszer – számítógép közti adatátvitelt soros vonalon (RS-232) keresztül valósítottam meg. Ezen összeköttetési kapcsolatot a következő jellemzőkkel lehet bemutatni (az egyes tulajdonságok melletti számértékek a munkáim során használt értékeket reprezentálják):

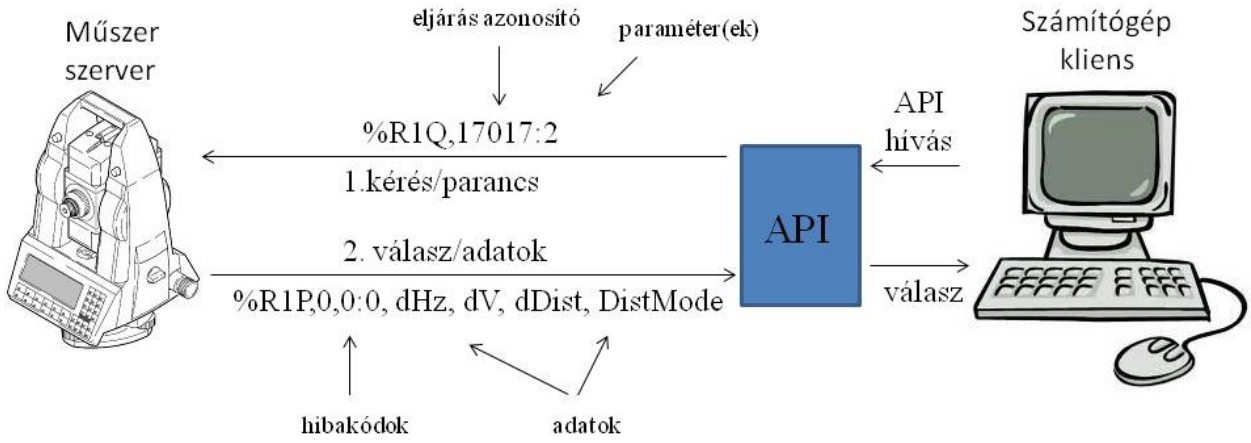
- sebesség: 9600 bit/s – Az adatátviteli sebesség megadja a másodpercenként áthaladó bitek számát;
- adat bitek: 8 – Egymás követő bitek száma, melyek egy információ egységet alkotnak;
- stop bitek: 1 – Az adatbitek egyes csoportjainak a kezdetét és végét megadó jelző bit;
- paritás: N ellenőrző bit az adatátviteli hibák észlelésére.

A műszer – számítógép közötti kommunikációra két féle eljárást alkalmaztam a mérések és tesztelések során. Az első módszer egy közvetlen kapcsolatkialakítás, amelynél a műszer vezérlése közvetlenül a gyártók által biztosított parancsokon keresztül történik. Ebben az esetben a lehetőségeink korlátozottak, bár az alapvető műveletek végrehajtására (pl.: távmérés egy prizmára, műszert második távcsőállásba való forgatása, stb.), valamint tesztelések, kísérletezések elvégzésére ezt is tökéletesen lehet alkalmazni. Az itt említett kapcsolati kialakítást a következő ábra szemlélteti:



2-5. ábra Műszer számítógép közti kapcsolat 1.(Leica)

A műszer számítógépről történő automatizált vezérlésnek igazi megoldása egy vezérlő program készítése. A gyakorlati alkalmazásaim során egy ilyen vezérlő API-t teszteltem és fejlesztettem, mely tartalmazta a műszervezérléshez szükséges alapvető eljárásokat és függvényeket. Ezen kapcsolatkialakítási mód már egy magasabb szintű megközelítés, sokkal kiterjedtebb a felhasználó által igénybe vehető lehetőségek halmaza. Itt fontos megemlíteni, hogy közvetlenül, egy adott műszerhez tartozó protokollt használva, műszer csere esetén módosítani kell a kódunkat, azonban egy API segítségével ki lehet alakítani többféle eszközt támogató rendszert is. Szakdolgozatom során felhasznált TclAPI programnyelve gép és platform független., egyaránt használható Windows XP/7 és Linux operációs rendszereken is. A műszer – számítógép egy API-n keresztül történő kapcsolatkialakítását a következő kép illusztrálja:



2-6. ábra Műszer számítógép közti kapcsolat 1.

Ebben a fejezetben részletezett programozási technikák alkalmazásával képesek vagyunk az robot mérőállomások működését irányítani, valamint a műszerek mérésének összehangolását elvégezni. Az ebben szereplő elemi műveletek az építőkövei a mozgásvizsgálati rendszereknek.

3. Automatizált mozgásvizsgálati mérőrendszerek

3.1. Mozgásvizsgálatok feladata

Egy folyamatosan változó dinamikus világban élünk. A körülöttünk lévő építőmérnöki szerkezetek, épületek, hidak, metró illetve vasút alagutak, gátak, bányakialakítások állandó mozgásban (egy illetve többirányú mozgások, rezgések, hajlítások) vannak, melyeknek meghatározása geodéziai méréseket igényel. Az eltérő rendeltetésű létesítmények mozgásai külső és belső erők hatására következhetnek be. Külső erőknek nevezzük a használatból adódó statikus és dinamikus terheket valamint az időjárási körülményekből és a közvetlen és közvetett tömegmozgásokból adódó hatásokat. A belső erők viszont a szerkezetek anyagait jellemző fizikai és kémiai tulajdonságok megváltozásaiból vezethetők le.

A mozgásvizsgálatok során a feladatunk az, hogy meghatározzuk különböző mérnöki létesítmények és objektumok egyes pontjainak egy adott viszonyítási rendszerben való elmozdulását, valamint az egyes szerkezeti elemek alakváltozását a rájuk ható erők hatására. A különféle deformációk és helyzetváltozások megfigyelése során a célunk sokrétű lehet, azonban főként az elmozdulások előrejelzésére és konkrét meghatározása a fő feladat.

A különböző objektumok mozgását összetett fizikai folyamatok írják le. Annak érdekében, hogy az egyes folyamatokat, észlelni, mérni tudjuk, valamint hogy a mérési eredményeket kezelhető matematikai formában kiértékelhetővé tegyük, különféle egyszerűsítésekkel, egyszerűsítő feltevésekkel kell élnünk. Az első feltevés során azt kell definiálnunk, hogy a vizsgált létesítményt az előre megjelölt pontjaival helyettesítjük. Így ennek következtében a pontok mozgásából levont konzekvencia a vizsgált objektumra lesz jellemző. Másodszor pedig azzal az egyszerűsítő feltételezéssel kell élnünk, hogy méréseinkhez felhasznált alappontok mozdulatlanok, így egyértelműen az adott objektum mozgására következtethetünk egy vizsgálat során.[ÉPMO]

3.2. Mozgásvizsgálati rendszerek jellemzése és feladata

Az összetettebb, nagyobb kiterjedésű szerkezetek monitoring vizsgálatához egyetlen robot mérőállomás nem mindig elegendő. Ahhoz, hogy az ilyesfajta építményekhez kapcsolódó mozgásvizsgálati feladatokat sikeresen végrehajtsunk, több programozható mérőállomásból összeállított mérőrendszerre van szükség. Az automatizált mozgásvizsgálati rendszer (Automatic Deformation Monitoring System – ADMS) egy olyan, egymással kölcsönhatásban, összefüggésben lévő, valamint egymástól kölcsönösen függő hardver és szoftver együttest jelent,

mely kiterjed egy mozgásvizsgálati feladat teljes megoldására, a mérőműszerek telepítésétől egészen az eredmények kiértékeléséig és továbbításáig. A mozgásvizsgálati mérőrendszerek létjogosultsága főként onnan ered, hogy az egyes mérések eredményeit valós időben (real-time) kell kiértékelnünk, veszély esetén pedig a deformációt követően minél hamarabb kell értesíteni az erre illetékes személyt. Az automatizált mozgásvizsgálati rendszerek egyik nagy előnye, hogy az elemzések az interneten is elérhetővé válhatnak a felhasználók számára, aki különböző jogosultságokkal rendelkezhetnek, ezzel kialakítva egy több felhasználós monitoring rendszert. A mérőrendszerek másik nagy előnye abban rejlik, hogy a biztonság és kármegelőzés érdekében különböző riasztóberendezésekkel látják el őket. A mozgásvizsgálati rendszerek szerves részét képezi a szoftver, segítségükkel a robot mérőállomások adatain kívül különféle geodéziai, geotechnikai és meteorológia szenzorok mérési adatait is fel tudja dolgozni, valamint a különböző elemzések (pl.: grafikonok készítése, eltérések kimutatása) végrehajtását is ez a szoftver végezheti. Az ADMS-es által szolgáltatott adatok homogenitása érdekében mozdulatlan referencia prizmák megléte szükséges a műszerek tájékozásához és az álláspont mozdulatlanságának vizsgálatához. A pontos (milliméter nagyságrendű) mérés végrehajtásához viszont vizsgálati prizmák létesítését, illetve állandósítását kell megoldani egy adott munka során.

3.2.1. A SolData által kifejleszett Cyclops és Centaur mozgásvizsgálati rendszer jellemzése

A SolData nevű francia cégről bátran ki lehet jelenteni, hogy világon az egyik legnagyobb tapasztalattal rendelkező, mozgásvizsgálati rendszerek fejlesztésével és telepítésével foglalkozó nagyvállalat. Számos világvárosban lévő (pl.: Barcelona, London, Budapest) metróvonal hálózat kiépítésének a biztonságáért ők feleltek, a nagy pontosságot és hatékonyságot biztosító mozgásvizsgálati rendszerüknek a működtetésével. A SolData feladata kiterjed a geotechnikai, szerkezeti, környezeti mozgásvizsgálati folyamatok ellenőrzésén keresztül egészen a hidrogeológiai monitoring vizsgálatokig. Nevükhöz fűződik a **CYCLOPS** és **CENTAUR** kifejezés, melyek az általuk kialakított automata mozgásvizsgálati mérőrendszerek megnevezése.

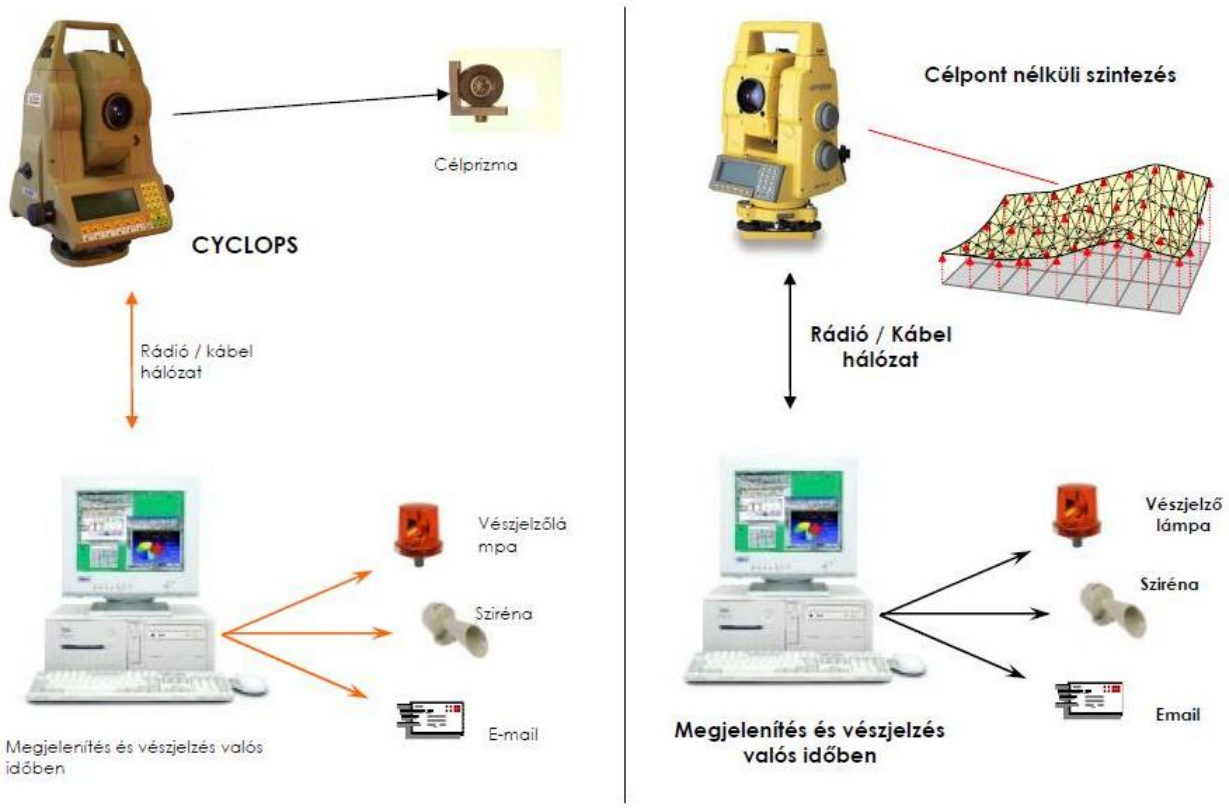
A SolData és az IGN (Institut Géogprahiqe Nationa), a Francia Országos Földmérési Intézet, közös fejlesztése révén készült el a **CYCLOPS** névre hallgató mérőrendszer. A CYCLOPS alapjában véve, egy teljesen automatizált geodéziai mérőrendszer, amely több számítógép által vezérelt, motorizált mérőállomásokból álló mozgáskövető rendszert jelent. Ezen műszer összeállítás olyan modern mérőállomásokkal van felszerelve, melyek lehetővé teszik a nagy

teljesítményű, sokoldalú, és felhasználóbarát monitorig-megoldás létrehozását. A rendszer a hét 6+minden napján, napi 24 órában megállás nélkül üzemel, ezzel biztosítva az egyes mérőállomásokhoz rendelt vizsgálati prizmák rendszeres bemérését. Ez gyakorlatilag azt jelenti, hogy az egyes robot mérőállomások, két távcsőállásban végigmérik az előre meghatározott irányzási sorrendben lévő, és a feladatkörüknek megfelelő a prizmákat, majd ezt a periódust megismételve újból elkezdik a méréssorozatot. A CYCLOPS tehát minden pontról abszolút mérési adatokat szolgáltat 3D-ben (X,Y,Z) a felhasználók számára, ezzel biztosítva az mérések valós időben történő kiértékelését, elemzését. A 3D-s adatszolgáltatás során kapott X,Y koordináták a vízszintes mozgás változásait, míg a Z koordináta a függőleges tengely menti elmozdulások értékeit adja meg. Ami a rendszer gyorsaságát illeti, a mérés prizmánként 12 másodperc, bár nem ritka a 3 másodperc (pl.: 4-es metró monitoring vizsgálata) alatt elvégzett mérési időtartam se. A rendszer sajátossága, amellett hogy éjjel-nappal üzemel, az hogy világítás nélkül is elvégzi az automata méréseket. A pontosságáról elmondható, hogy 60 méteren kb. +/- 0,5mm, a hatótávolsága viszont az épülő 4-es metró vonalán 500m-ig terjed TCA 1800-as műszerek esetén. A rendszer által szolgáltatott információ esetenként kiegészülhet manuális mérések eredményeivel, azonban ezen mérések már más, azonban egymással kapcsolatban lévő hálózatot képviselnek. Mindent összevetve, a CYCLOPS alkalmazása által az építőmérnökök és a kivitelezők számára lehetőség nyílik szerkezetek deformációinak és helyzetváltásainak a valós idejű megfigyelésére és ellenőrzésére egy adott munkaterületen.

A fent említett két intézetnek egy másik, szintén közös fejlesztése a **CENTAUR** elnevezésű mozgásvizsgálati rendszer, mely magasságmeghatározási metódust alkalmazó automata robot mérőállomások segítségével, egy modern, valós idejű monitoring módszert tesz lehetővé. A CENTAUR egy teljes mértékben automatizált mérőrendszer, alkalmazása lehetővé teszi felületek automatizált mozgásvizsgálatát. A rendszer legfőbb sajátossága, hogy a méréseket prizma nélkül végzi, ennek következtében prizmák előzetes állandósítása nélkül folyhat a mozgásvizsgálati munka. Előnye a CYCLOPS-éhoz hasonlóan, hogy a felület megvilágítása nélkül működik a hét minden napján, 24 órában. A CENTAUR abszolút függőleges alakváltozásokat mér, melynél a mért pontok egy virtuális vízszintes hálón helyezkednek el. A mérőműszer a vízszintestől számított 10-60°-os tartományban végzi el a deformációk mértékének a meghatározását, a tájékozást pedig külső referenciapontok segítségével hajtja végre. A rendszer tehát süllyedések és emelkedések valós idejű megfigyelésére szolgál, valamint elmondható róla, hogy bevált rendszerként ismerik és használják világszerte. Ami a mérés

pontosságát illeti, +/-0,5mm 40m-en, bár ez erősen függ a felület minőségétől (beton, aszfalt, homok, stb.), valamit a referencia pontok stabilitásától is.

Gyakorlati alkalmazásai között főként az autópályák, repülőterek fel- és leszálló pályáinak valamint hidak felületének az automatizált vizsgálata szerepel.



3-1. ábra CYCLOPS és CENTAUR mozgásvizsgálati rendszer kialakítása [SOLD]

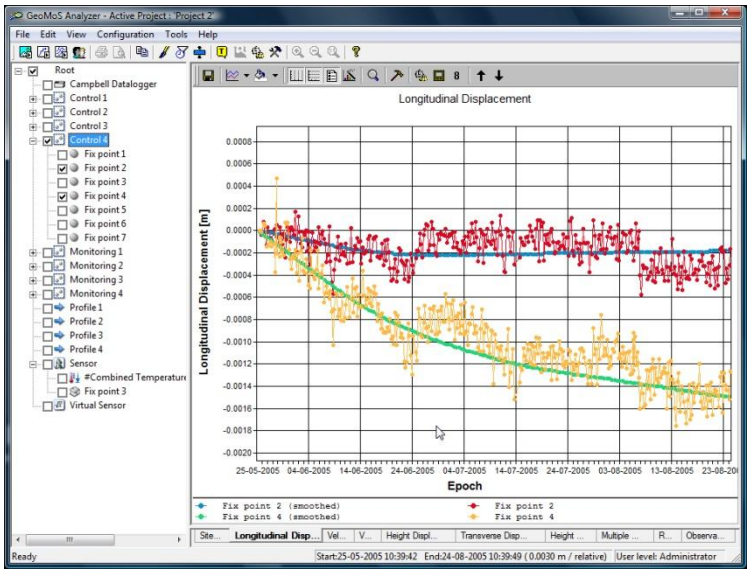
Mindkét említett rendszerről elmondható, hogy az egyes mérőműszereket védőrács védi a sérülésektől, ezzel megóvva ezeket az esetleges károktól. A CENTAUR összeköthető a CYCLOPS rendszerrel, illetve a két mérőrendszer egymást kiegészítve is működhet, ezzel a szerkezetet mozgásvizsgálatának hatékony és modern technológiát kínálva.

Az két egymástól különböző, azonban összeköthető mozgásvizsgálati rendszer által szolgáltatott adatok beolvasásáról és vizuális megjelenítéséről valamint az ezekből származó információk továbbításáról a SolData által fejlesztett **Geoscope Web** nevezetű elemző szoftver gondoskodik. A program egy GISWeb-nek (Geographic Information System Web) felel meg, amely interneten működő térinformatikai rendszert jelent. A Geoscope webes felülete bárki számára elérhető, aki rendelkezik internet kapcsolattal, azonban a rendszerbe való belépés jogához csak az adott munkához tartozó mérnökök, kivitelezők és természetesen a beruházó juthat. A szoftver fő

feladata a CENTAUR és CYCLOPS rendszernél használt mérőműszerek adatainak, valamint az említett rendszereket kiegészítő különféle szenzorokból származó (pl.:meteorológiai szenzorok) adatok valós idejű feldolgozása és kiértékelése. A Geoscope Web működését különféle szolgáltatásokkal lehet kiegészíteni, melyek közül a legfontosabb a riasztás. Ennek alkalmazásával egy olyan riasztási rendszert lehet működtetni egy adott munkafolyamat során, melynek segítségével különböző riasztási szinteknek (Első-, Másod-, ill. Harmadfokú riasztási szint) megfelelően lehet jelezni az építés közben felmerülő veszélyt a mérnökök és kivitelezők felé.

3.2.2. Leica GeoMoS

A Leica Geosystems a világ egyik legnagyobb, földméréssel és földrajzi helymeghatározással kapcsolatos műszereket és szoftvereket forgalmazó piavezető vállalata. Az anyavállalat Svájc keleti részén helyezkedik el, azonban szinte az összes kontinensen forgalmazza a termékeit, Magyarországot is beleértve egészen Kináig. A különféle eszközök és szoftverek fejlesztése, gyártása során a legújabb technológiai találmányokat építi be, melyek hatékonyan képesek kielégíteni az automatizált rendszerekkel szemben támasztott követelményeket. Az automata mozgásvizsgálati mérőrendszerek terén **a Leica GeoMoS**-t (Geodetic Monitoring Software) világszerte bevált monitoring szoftverként ismerik. Alkalmazásával a legkülönfélébb mérnöki létesítmények (pl.: felhőkarcolók, gátak, alagutak) mozgásvizsgálata válik lehetővé.



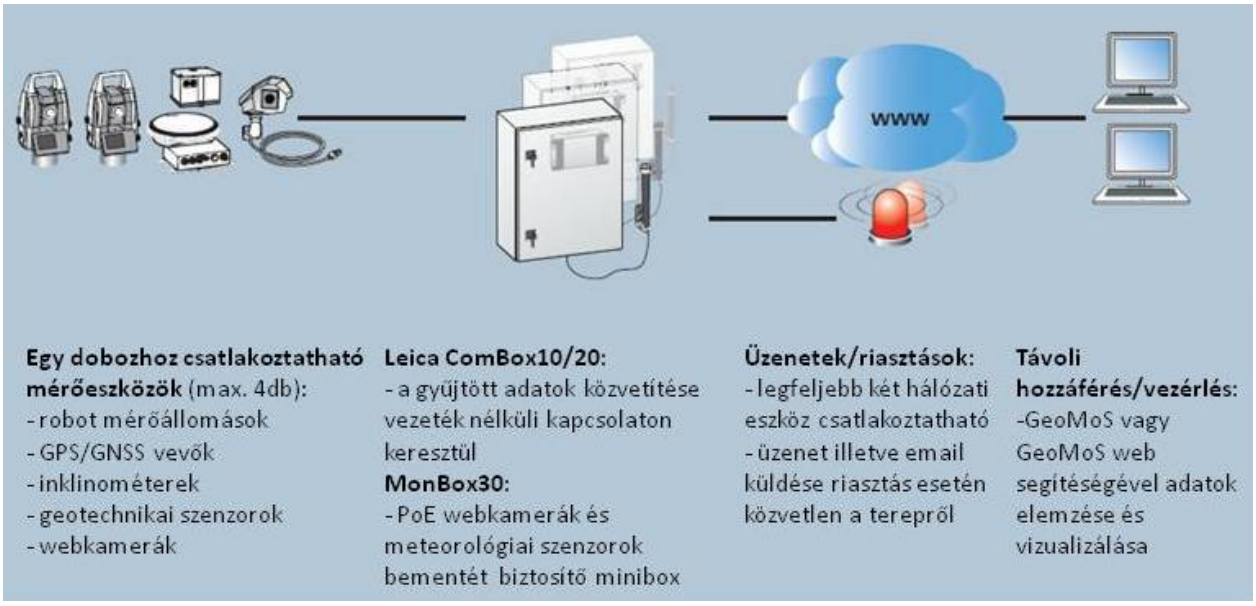
3-2ábra Leica GeoMoS szoftver – Analyzer [LEIC]

A GeoMoS szoftver magában foglal két fő alkalmazást, és egy kiegészítőt (add-on) is, melyek a következők:

- **Monitor:** Ez egy online alkalmazás, mely az egyes mérőműszerek és szenzorok irányításáért, adatok összegyűjtésért és az adott munkafolyamat szervezéséért (event management) felelős

- **Analyzer:** Az Analyzer egy offline alkalmazás, melynek köszönhetően az egyes elemzések készítése, az eredmények megjelenítése valamint az adatok utófeldolgozása válik lehetővé.
- **Adjustment** add-on**:** Ezen kiegészítő segítségével a hálózati rendszer beállítása, és szimulálása, valamint a deformáció analízis végezhető el. [LEIC]

A program nagy előnye, hogy szinte az összes modern műszerrel kompatibilis, ezzel biztosítva a nagyfokú teljesítőképességet. A GeoMoS-nak a többi műszerrel és szenzorral való kommunikációját a szokásos kapcsolatlétrehozási módszerek mellett, a Leica saját fejlesztésű **Leica M-COM** rádiós hálózat segítségével könnyedén és rugalmasan megoldható. A M-COM egy olyan kommunikációs csatornát létrehozó „dobozból" (ComBox10/20) áll, mely az adatoknak a világháló és ezen keresztül a GeoMoS felé történő továbbításáért felel. A ComBox legújabb változatai tartalmaznak egy MonBox nevezetű „minibox"-ot is melynek köszönhetően a meteorológiai szenzorok és a PoE (Power over Ethernet) webkamerák csatlakozása is lehetővé válik. Az M-COM által kialakítható mozgásvizsgálati rendszerhálózatot a következő kép szemlélteti:



3-3 ábra Leica M-COM segítségével kialakítható rendszerhálózat [LEIC]

3.3. Munkagépek vezérlését lehetővé tevő vezérlő rendszerek

3-4. ábra 360°-os prizmával felszerelt munkagép a terepen [GEPN]

Az építőiparban használatos földmunka- és építőipari gépek vezérlése is irányítása különböző igények szerint alakítható ki. A GPS segítségével történő globális helymeghatározás széles körű elterjedésével a műholdas gépvezérlés kivitelezhető vált a mérnökök számára. Ezzel a megoldási módszerrel a munkagépre szerelt antenna pozícióját határozhatjuk meg, majd ez ebből származó, munkagép helyzetét leíró adatok segítségével tudjuk az anyagmozgató gépek hidraulikus nyomással működő alkatrészeit (pl.: tolólap vagy vágóél) – vízszintes és magassági értelemben – a helyes útra irányítani. Ehhez természetesen nélkülözhetetlen egy számítógép a munkagép fülkéjében. A pontos helyzet meghatározásának érdekében a terepen GPS bázisállomást kell létesítenünk, ennek hiányában pedig GPS referencia hálózat megléte szükséges.

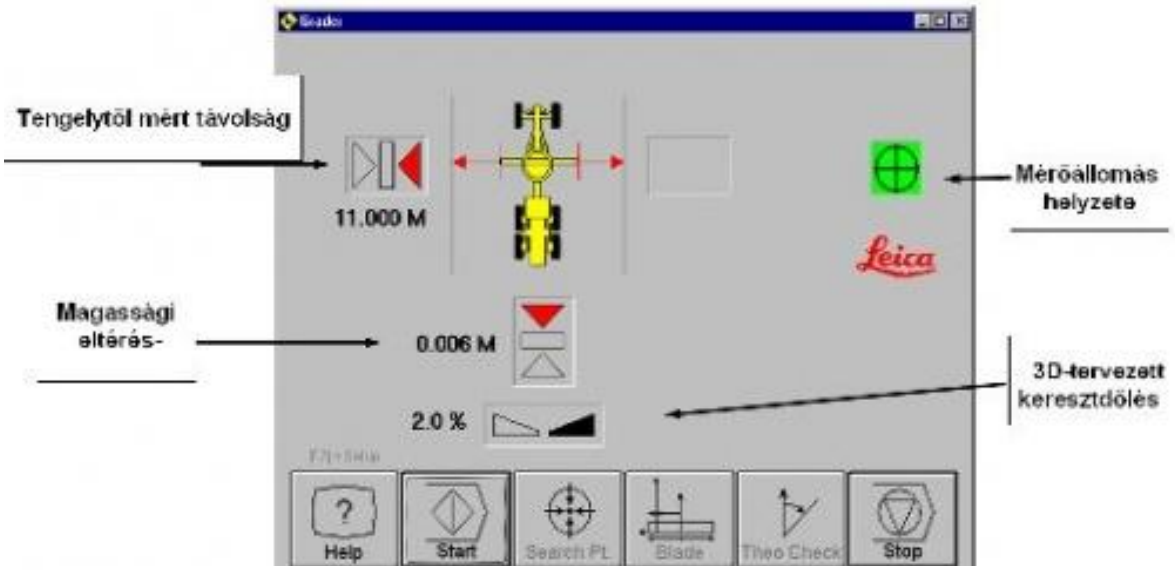
A 3D-ben történő földmunkagép vezérlés a robot mérőállomások alkalmazása szintén lehetővé teszi. Ez esetben a mérőállomás a munkagépre szerelt 360°-os prizmára végzi a pozicionálást, majd rádiójelen keresztül továbbítja az adatokat a fedélzeti számítógép felé. Bár az egyes munkagép vezérlési folyamatok a céljukat tekintve sokrétűek lehetnek (legyen szó felület kialakításról, tömörítésről, vagy egy munkagépnek vízszintes értelemben vett irányításáról), azonban ahhoz, bármelyik feladatnál sikeresen lehessen alkalmazni a mérőállomások által nyújtott gépirányítási módszert, különféle feltételeknek kell eleget tenni. Elsőként is a mérőműszert egy ismert alappont-hálózathoz kell kapcsolnunk, melyet a szükséges tájékozás elvégzésével tudunk véghezvinni. Másfelől pedig biztosítanunk kell a mérőállomás és a prizma permanens összeláthatóságát. A munkagépeknek mérőállomásokkal történő 3D vezérlését, irányítását és mozgásvizsgálatát a könnyebb áttekinthetőség végett két részre lehet osztani:

- Magassági értelemben vett monitoring: Ez esetben a fedélzeti számítógépnek tartalmaznia kell a munka során kialakítandó, avagy követendő terepmodellt. Ilyenkor a kialakított rendszer a terepi és a tervből származó magassági adatok alapján vezérli a munkagép anyagmozgató eszközének a magassági értelemben vett mozgatását. A terep- illetve domborzatmodell elkészítésére sokféle technológia között választhatunk, kezdve a

raszteres térképekből való levezetéstől egészen a modern lézerradarok által nyújtott módszerekig. Bármelyik metódust is használjuk, ezeket még a földmunka elvégzése előtt, előzetes terepi, irodai és számítógépes munka keretén belül kell elvégeznünk. A modell feltöltését követően pedig, a munka kezdetét veheti.

- Vízszintes értelemben vett monitoring: A vízszintes értelemben vett irányítás illetve ellenőrzés tekintetében nincs szükségünk magassági adatokra. Ekkor csak a követendő, avagy kialakítandó nyomvonal koordinátalistáját kell feltöltenünk a munkagép számítógépére, majd a mérőállomás által bemért adatokból és a feltöltött útvonal koordinátáiból számíthatók az adott munka elvégzéséhez szükséges paraméterek. A vízszintes értelemben vett irányítást, illetve ellenőrzést lehetővé tevő rendszerek alkalmazásával a felhasználónak lehetősége nyílhat például egy híd betolásának az irányítására, avagy különböző sínpályák ellenőrzésére.

Az itt említett két különböző, bár egymással összeköthető mérési variáns megoldásait tekintve szerte ágazó lehet. A nevesebb geodéziai cégek (pl.: Leica Geosystem, Topcon) egytől egyik rendelkeznek saját készítésű szoftverrel, melyek funkcionalitásukat tekintve különbözőek lehetnek, bár van mód saját készítésű rendszer és egyben szoftver készítésére is. Egy saját készítésű, vízszintes értelemben vett mozgásvizsgálati program elkészítését az 5. fejezet, Gyakorlati alkalmazások alfejezete tárgyalja.



3-5. ábra Leica Geosystem - Munkagép irányító szoftver [LEIC]

3.4. SolData megoldások hazánkban

3.4.1. 4-es metró építésének monitoring vizsgálata

Az elmúlt évek legnagyobb volumenű építkezése a 4-es metró építése, melyhez kapcsolódó munkálatok a mai napig tartanak. A metró Budapest viszonylag nagy területét szeli át, ahol – a geológiai vizsgálatokat elemezve – kiderült, hogy a talajok különböző fizikai tulajdonságokkal rendelkeznek, valamint hogy az építkezés sűrűn beépített területek alatt halad. Ennek következtében szükségszerűvé vált a metróépítés felszínre gyakorolt hatásának a vizsgálata és nyomon követése továbbá egy zaj- és rezgésellenőrző rendszernek a kiépítése, mely a SolData S.A.S. nevű francia cég hozzájárulásával valósult meg. Az állomások környezetébe lévő épületek, épületegyüttesek hagyományos módon végzett mozgásvizsgálatát a Hungeod Kft. hajtja végre. Az ily módon bemért adatok kiegészítő, de egyben nélkülözhetetlen adatként fognak szolgálni az automata mozgásvizsgálati rendszer eredményei mellett.

3-6. ábra Cyclops: Mérőállomás elhelyezkedése [SOLD]

A SolData-val üzleti kapcsolatban lévő Soldata Hungeod Konzrocium feladata volt a metróvonal I. építési szakaszán az automatizált mozgásvizsgálati rendszer kiépítése és beüzemeltetése. A munkájuk úttörő tevékenységnek számít, mivel Magyarországon először mutatkozik be a saját CYCLOPS és CENTAUR rendszer. Ezen két (3.2.1. fejezetben részletezett) mérőrendszer egymástól függetlenül, de egymást kiegészítve működik a megbízás során, napi 24 órában, a hét minden napján. A CYCLOPS hatékony működését 30 robot mérőállomás, és több mint 1500 monitoring prizma teszi lehetővé, melyekről kb. fél óránként tud friss adatot szolgáltatni a rendszer. Ezeken kívül az eredmények kiegészítése és javítása érdekében egy-egy meteorológiai mérőállomás működik Pesten és Budán, továbbá manuális rezgésmérések is folynak az építés által igénybevett területeken. A CENTAUR névre hallgató, süllyedések és emelkedések meghatározását lehetővé tevő monitoring rendszer eredményességéhez viszont 120 automata szintezőműszer 50 extenzométer, 50 piezométer valamint 20 inklinométer járul hozzá. [MTMM]

3.4.2. Ajkai vörösiszap-katasztrófa utáni mozgásvizsgálat észlelése

3-7. ábra Ajkai vörösiszap-katasztrófa utáni mozgásvizsgálati munkálatok [SOLD]

Az ajkai timföldgyár vörösiszap tározójának átszakadása 2010.-ben Magyarország eddigi legnagyobb ökológiai következménnyel járó ipari katasztrófája. A tragédia során kiömlő vörösiszap, több mint 800 hektárnyi területet árasztott el, több emberéletet követelt, valamint felbecsülhetetlen környezeti és anyagi károkat okozott. A katasztrófa bekövetkezését követően az Országos Katasztrófavédelmi Főigazgatóság a SolData céget bízta meg azzal, hogy építsen ki egy automatizált mozgásvizsgálati rendszert a helyszínen annak érdekében, hogy a gátfal további kritikus mozgásait mihamarabb észlelni lehessen. A SolData csapata a feladat elvégzését egy Cyclops rendszer telepítésével és működtetésével oldotta meg, melyekből származó valós idejű adatokat 6 darab automata dőlésmérő is segítette. Az adatok továbbításáról rádiós kapcsolat gondoskodott, az egyes elemzések eredményeinek megjelenítése a felhasználók számára (mérnökök) pedig a Geoscope Web online szoftver által vált elérhetővé. Annak érdekében, hogy a még felmerülő kritikus mozgások bekövetkezésekor az esetleges szükséges intézkedéseket még időben el lehessen végezni, egy riasztási rendszer kiépítése vált elengedhetetlenné. [SOLD]

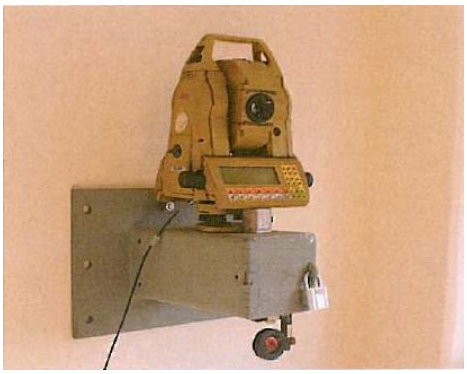
3.5. Külföldi mozgásvizsgálati rendszer a Leica Geosystem támogatásával: Honkong-i vasútvonal kibővítésének monitoring vizsgálata

Hong Kong-ban egy fejlesztési beruházás keretein belül, egy közel 1.2 km hosszú, új vasútvonal megépítést tervezték, a már meglévő Airport Express Line (Reptéri Expressz Vasútvonal) közvetlen közelében. Az építési munkálatok zavartalan lefolyása érdekében nélkülözhetetlen volt egy automatizált mozgásvizsgálati rendszer (ADMS) kiépítése. A rendszer létesítésére a Leica Geosystem-et bízták meg, mivel a részvénytársaságként is ismert vállalat számos tapasztalattal és referencia munkával rendelkezik a vasúti vonalak mozgásvizsgálati ellenőrzését illetően. Ennek következtében a Leica biztosította projekt során felhasznált összes műszert, szenzort és szoftvert, beleértve ezeknek a telepítését, tesztelését, üzembe helyezését és az eredmények megjelenítését az ügyfelek számára.

A kiépítéssel kapcsolatos munkálatok 2006. január első hetében kezdődtek, és majd 8 hetet vett igénybe a teljes rendszer implementálása. Ezt követően az ADMS 36 hónapon keresztül, napi 24

órában figyelemmel kísérte az építési terület deformációs folyamatait, vizsgálva a környező vasúti vonalak hatását a helyszínre. Az előírásoknak megfelelően a pályát három szakaszra (Ballasted Section - 600m, Trough Section – 210m, Tunnel Section – 400m) ++ osztani a hatékony monitoring vizsgálat érdekében. A 3 szakasz vizsgálatához 14 darab TCA2003-as robot mérőállomást és összesen 560 prizmát használtak fel. A műszerek, prizmák és a felhasznált szoftverek számának az eloszlását a következő táblázat írja le:

Szakasz	Vizsgálati prizma	TCA2003	GeoMoS
Tunnel	60db	4db	1db
Ballasted	370db	6db	2db
Trough	130db	4db	1db

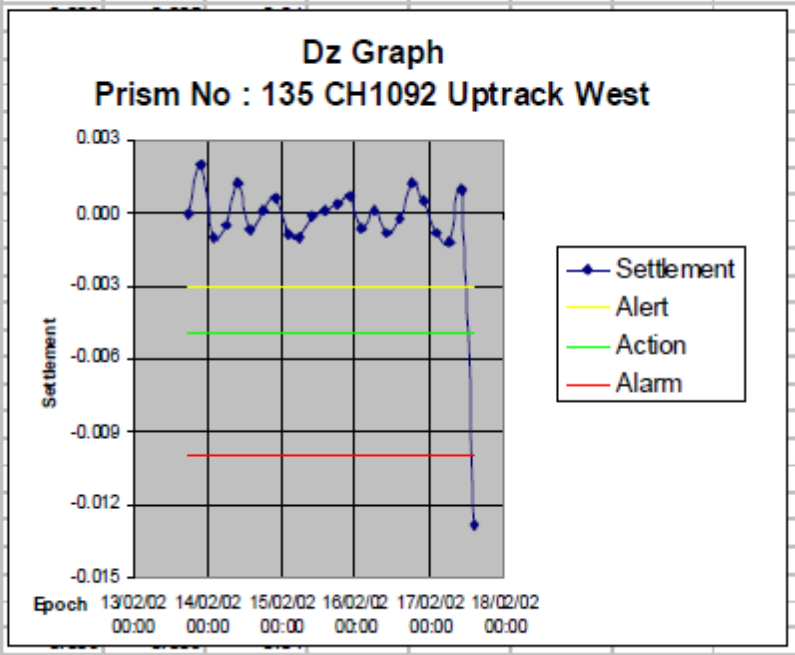


3-8. ábra TCA2003 az alagút szakaszban

A hardverkomponensek fő részét képező TCA2003-as műszerek számítógép által vezérelt robot mérőállomások, melyek +-0.5”-es szögmérési továbbá 1mm+/-1ppm-es távmérési pontosságukkal biztosították a prizmák szabatos mérését. A műszerek elhelyezését illetően az egyes szakaszokban különbözően jártak el. Az első, alagút szakaszban (Tunnel section) a mérőállomásokat a falakra szerelték fel, míg a 2. szakaszban (Ballasted Section) pillérek segítségével helyezték el a mérőműszereket. Az adatátvitel és az áramellátást vezetékes átviteli technológiával oldották meg a központi vezérlő és a mérőállomások között. A pontosság folyamatos fenntartása érdekében 6 havonta kalibrálták és karbantartották az összes műszert beleértve a szervomotorok olajozását és esetleges javítását minden alkalommal. A tájékozások érdekében természetesen referencia prizmákat létesítésére is szükség volt a süllyedési zónákon kívül.

A létesített automatizált rendszer magját azonban a GeoMoS monitoring szoftver képezi, melynek a bővebb jellemzése **3.2.2.** fejezet alatt található. A szoftver elemi részét képezi a Leica Sensor Manager, mely különböző típusú szenzorok által szolgáltatott adatok kiolvasásáért és az eredmények elemzéséért szolgál. A GeoMoS szoftver alkalmazásával az mérőállomások irányításán túl lehetőség nyílik a hőmérséklet és meteorológiai adatok integrálására, ezzel javítva a mérés pontosságát. Annak érdekében, hogy az adatok a felhasználók számára is elérhetővé váljanak, a szoftver továbbküldi őket egy webes felületre (GeoMoS Web). A GeoMoS szolgáltatásai közé tartozik a riasztás is, amelynet akkor aktiválnak, amikor egy adott vizsgálati pont elmozdulása egy adott határértéket túllép, emellett pedig két óránként egy jelentés (report) fájlt készít a szoftver, melyet a meghatározott személyeknek e-mail-ben azonnal elküldi. A 135-ös számú vizsgálati prizmának, egy adott időtartamhoz kapcsolódó report-ját a következő fájl szemlélteti:

Prism No:	135				
Location	CH1092 Uptrack West				
Epoch	Z-value	Settlement	Alert	Action	Alarm
13/02/02 18:00	1.3550	0.0000			
13/02/02 22:00	1.3570	0.0020			
14/02/02 02:00	1.3560	-0.0010			
14/02/02 06:00	1.3555	-0.0005			
14/02/02 10:00	1.3567	0.0012			
14/02/02 14:00	1.3560	-0.0007			
14/02/02 18:00	1.3561	0.0001			
14/02/02 22:00	1.3568	0.0007			
15/02/02 02:00	1.3559	-0.0009			
15/02/02 06:00	1.3549	-0.0010			
15/02/02 10:00	1.3548	-0.0001			
15/02/02 14:00	1.3549	0.0001			
15/02/02 18:00	1.3553	0.0004			
15/02/02 22:00	1.3560	0.0007			
16/02/02 02:00	1.3554	-0.0006			
16/02/02 06:00	1.3555	0.0001			
16/02/02 10:00	1.3547	-0.0008			
16/02/02 14:00	1.3545	-0.0002			
16/02/02 18:00	1.3557	0.0012			
16/02/02 22:00	1.3562	0.0005			
17/02/02 02:00	1.3554	-0.0008			
17/02/02 06:00	1.3542	-0.0012			
17/02/02 10:00	1.3552	0.0010			
17/02/02 14:00	1.3423	-0.0129			



3-9. ábra Email-ben kiküldött report fájl [FIGN]

Mindent összevetve a 3 év alatt összesen 7 millió adatsor gyűjt össze, naponta ez 6720 feldolgozandó adathalmazt jelent, melyek a központi szerveren lévő adatbázisba tároltak. A Leica által biztosított AMDS-ről tehát elmondható, hogy egy nagyon rugalmas rendszerről van szó, mely képes egy vasút- illetve alagútvonal valós idejű monitoring vizsgálatára, kiegészítve az adatokat különféle szenzorok adataival, valamint biztosítva a mérnökök számára azt a lehetőséget, hogy az esetleges beavatkozási munkákat időben megtegyék ha erre szükség van.

4. Mérőfelszerelések vizsgálata

4.1. Az egyes műszerek kalibrálása

A különböző geodéziai mérésekhez használt műszerek és eszközök pontossága és megbízhatósága kulcskérdés. A mérőműszerek megvásárlása után is gondoskodni kell arról, hogy az általuk mért értékek a mérendő mennyiség valódi értéke közötti eltérés a műszerre jellemző megbízhatóságánál ne legyen nagyobb. Erre szolgál a mérőeszközök, ezen belül is a mérőállomások zöménél a kalibrálás, amelyet időről időre el kell végezni, ha biztosítani kívánjuk a műszerünk által mért értékek elfogadhatóságát.

A gyakorlati alkalmazásaim során használt Leica robot mérőállomások (TCA1800, TPS1201) olyan beépített kalibrációs programmal (Check & Adjust) rendelkeznek, melynek segítségével az egyes műszerhibák meghatározhatók. A műszervizsgálat menete hasonló alapokra épül, mint a Májay-féle műszervizsgálat, azonban itt az egyes műveletek elvégzése után a program műszerhibákat automatikusan kiszámítja. Ahhoz, hogy a mindennapi munkák során a legpontosabb eredményeket kapjuk, időről időre ajánlott a kalibrációt elvégezni, különösen akkor, ha egy nagyobb projekt végrehajtására készülünk. A mi esetünkben a vizsgálat célja a vízszintes és magassági szögmérés szabályos hibáinak meghatározása, melyek egyetlen távcsőállásban méréskor terhelik a körleolvasásokat.

A vizsgálat során, a későbbiekben felhasznált három darab Leica típusú robot mérőállomásnak határoztuk meg a kalibrációs értékeit, melyek a következők:

- l,t – kompenzátor hossz- és keresztirányú hibája;
- i – magassági kör indexhibája;
- c – a vízszintes szögmérés kollimáció hibája;
- a – fekvőtengely merőlegességi hibája;
- ATR – az ATR irányzásának a vízszintes és függőleges irányú hibája.

Az egyes, elektronikus módon igazított hibák összefoglaló táblázata:

Műszer hibák	Vízszintes szögmérésre való hatása	Magassági szögmérésre való hatása	Két távcsőállásban való kiejtési lehetőség	Automatikusan korrigálható a megfelelő beállítással
c - kollimáció hiba	✓	---	✓	✓
a - fekvőtengely merőlegességi hibája	✓	---	✓	✓
l - kompenzátor hosszirányú hibája		✓	✓	✓
t - kompenzátor keresztirányú hibája	✓	---	✓	✓
i - magassági kör indexhiba	---	✓	✓	✓
ATR - ATR irányzási hiba	✓	✓	---	✓

Az egyes műszerhibák meghatározását a Budapesti Műszaki és Gazdaságtudományi Egyetem R épülete előtt végeztük el, mivel itt voltak a műszerek kalibrálásához megfelelő körülmények adottak. A feladat végrehajtása során próbáltuk a kalibrálásra veszélyes hibahatásokat a minimálisra csökkenteni, a napnak az intenzív fényét eltakartuk, a nagy forgalmat pedig minden esetben kivártuk, hogy lecsendesedjen, mivel ezek befolyásolták volna a méréseinket, ezzel meghiúsítva a műszerek hibáinak a szabatos meghatározását. Elsőként felálltunk az R épülettel szembeni vasbeton pillérre, majd a tőle egy 100 méternél nagyobb távolságra lévő, masszív kőpárkányra, egy Kern pillértalp segítségével elhelyeztünk egy Leica kör prizmát. A vizsgálatunkat a Leica 1201-es műszer kombinált programjával kezdtük, elsőként meghatározva a vízszintes hibákat, majd az ATR vízszintes és magassági értelemben vett hibáját határoztuk meg. Ezt követően egy 27 és 63 fok közötti, jól megirányozható, R épületen lévő pontot irányoztunk meg két távcsőállásban, ezzel meghatározva az egyes magassági körön tett leolvasások hibáit is. Az 1201-es műszer kalibrálását elvégezve, a két darab TCA1800-as műszer hibáit is meghatároztuk. A TCA műszerek a kalibrálás kezdésénél automatikusan beállítja a kompenzátort hossz- és keresztirányú hibáit, így ezzel kapcsolatban mérést nem kellett végeznünk. Majd ezt követően pedig az a feladatrész következett, melynek keretén belül az *„i,c,a (magassági, kollimáció, fekvőtengely hibája)”* nevű kombinált vizsgálatot végeztük el, és végül befejezésül az ATR vízszintes és magassági hibáját határoztuk meg, ugyanazon a helyszínen, ahol a Leica 1201-es mérőállomással dolgoztunk. A beépített kalibráló programokon több ismétlést és elvégeztünk, hogy a lehető legpontosabb eredményekre jussunk. Az 1201-es műszer kalibrációs programja az egyes ismétléseknél meghatározott eredményeket átlagolta, az 1800-as műszereknél pedig manuálisan, papíron jegyeztük az eredményeket, majd ezek közül kiválasztottuk legmegfelelőbbet.

Az egyes műszerek kalibrációs beállításait a következő táblázatok mutatják:

Műszer neve : TCA 1800		
Serial Number : 270395		**Hibajavítások**
Hibák:	l	00-00-19
	t	00-00-11
	i	-00-00-05
	c	00-00-02
	a	-00-00-05

Műszer neve : TCA 1800		
Serial Number : 421157		**Hibajavítások**
Hibák:	l	00-00-16
	t	00-00-02
	i	-00-00-05
	c	-00-00-02
	a	00-00-00

Műszer neve : Leica 1201		
Serial Number :		**Hibajavítások**
Hibák:	l	-00-00-06
	t	00-00-01
	i	00-00-03
	c	00-00-14
	a	00-00-24
	ATR Hz	00-00-6

A fizikai távmérések során, a műszeren kijelzett távolságértékek a távmérő által mért távolság javított értéke. A javítás három fő összetevőből áll, melyek a következők: összeadó állandó, a szorzóállandó és a meteorológiai javítás. Az összeadó állandó a távmérés egyik

4-1. ábra: Összeadó állandó közvetlen úton történő meghatározása

legmeghatározóbb hibája, amely minden mérési eredményt ugyanolyan mértékben terhel. A hiba eredete egyaránt adódhat a műszer és a prizma felépítéséből is. A műszer esetén a hiba abból ered, hogy a távmérési jel kibocsátási pontja (elektromos nulla pont) nem esik egybe a műszer állótengelyével, ezt nevezzük műszer állandónak. A prizma estén pedig, a prizma visszaverődési pontja nem esik pontosan a vizsgálati pont függőlegesébe, ezt nevezzük prizmaállandónak. Ennek a két eltérésnek az összege az összeadó állandó. A meghatározására az egyik alkalmazható módszer egy távolság közvetlen és két részben végzett méréséből adódik. Ennél a metódusnál egy vízszintes távolságot mérünk meg egészében (D), majd a távot 1/3-2/3 arányban

elosztva lemérjük két részletben is (D_1, D_2). Így a következő képlet használatával az összeadó állandó könnyedén meghatározható: $c = D-(D1+D_2)$.

Létezik az összeadó állandó meghatározásának egy egyesszerűbb módja is, méghozzá az, hogy egy ismert hosszúságú, alapvonal hosszát mérjük meg a mérőberendezéssel, majd az alapvonal ismert hosszából és a mért eredmények különbségéből meghatározzuk az összeadó állandót.

A robot mérőállomásokkal végzett gyakorlati mérésekhez használt prizmáknak a prizmaállandóit a Bodola Lajos komparátor teremben határoztam meg. A mérőállomással és az egyes prizmákkal a teremben elhelyezkedő pillérekre álltam fel, szabatosan, Kern pillértalp segítségével. A Leica 1201-es műszer esetén a távolságmérés módját átállítottam „standard" módról „5 mérésből átlagoló"-ra, ezzel még nagyobb pontosságot biztosítva az állandók meghatározásához. A mérési eredményeket a következő táblázatok tartalmazzák:

Mini prizma(412)			
D1	**D2**	**D**	**Prizmaállandó**
4,999	17,383	22,390	0,008
4,999	17,383	22,391	0,009
4,999	17,383	22,390	0,008
5,000	17,383	22,391	0,008
4,999	17,383	22,390	0,008
A prizmaállandó értéke (átlagolás után) : **8,2mm**			

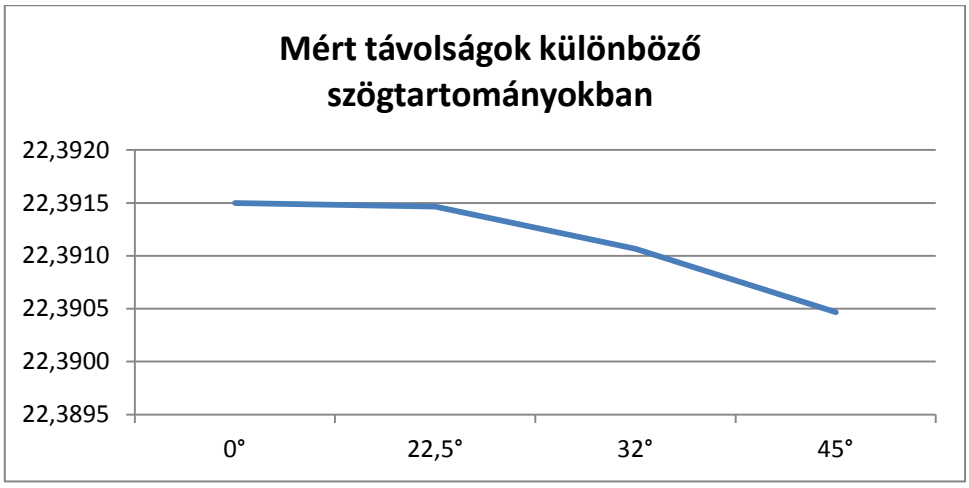
Leica 360° prizma			
D1	**D2**	**D**	**Prizmaállandó**
4,9836	17,3679	22,3745	0,0230
4,9835	17,3680	22,3746	0,0231
4,9835	17,3679	22,3744	0,0230
4,9836	17,3679	22,3745	0,0230
4,9835	17,3679	22,3745	0,0231
A prizmaállandó értéke (átlagolás után) : **23,0mm**			

Az egyes prizmaállandók meghatározása során kitértem arra az eshetőségre is, amikor az egyes prizmákra nem merőleges irányból, hanem attól eltérő, ferde irányból mérünk rá. A mini 412-es számú prizmára való ferde mérési eredményeket a következő táblázat foglalja össze:

Mini prizma(412)	1. mérés	2. mérés	3. mérés	**Átlag [m]**
0°	22,3915	22,3915	22,3915	**22,3915**
22,5°	22,3914	22,3915	22,3915	**22,3915**
32°	22,3911	22,3910	22,3911	**22,3911**
45°	22,3905	22,3904	22,3905	**22,3905**
55°	-	-	-	**-**

A mérési eredmények arra engednek következtetni, hogy 0°-tól 22,5°-ig a távolságmérésben nem mutatkozik eltérés, azonban 22,5°- 45°-ig viszont közel lineárisan trendet mutatnak a mért eredmények, tehát ebben a szakaszban lineárisan csökken a mért távolság. A táblázat

vizsgálatánál könnyen rájöhetünk arra is, hogy 55° és a felett a prizmára nem tudtunk távolságot mérni, így ezeket a gyakorlati méréseknél fontos figyelembe venni.



5. Ulyxes rendszer

5.1. A rendszer leírása

Az Ulyxes (Odüsszeusz) nevű projekt egy nyílt forráskódú eszközöket felhasználó mozgásvizsgálati rendszert foglal magában, melynek a szerzője Dr. Siki Zoltán. A projekt elsődleges célja, hogy egy olyan keretrendszert hozzunk létre, melynek a használatával képesek legyünk különböző típusú és gyártójú robot mérőállomások számítógépről történő vezérlésére, valamint a mért adatok interneten történő publikálására. A rendszer kialakításához számos nyílt forráskódú projekt hozzájárul, melyek a szerves részeit képezik az Ulyxes-nek:

Tcl: A Tcl (Tool Command Language) egy egyszerű, magas szintű, karaktersorozat (string) alapú szkript nyelv, melynek a megalkotója Jhon K. Ousterhout. A nyelv egy interpretált programnyelv, amely csak néhány alapvető konstrukcióval és viszonylag rövid szintaxissal rendelkezik így bárki számára könnyen tanulható, azonban a mindennapi egyszerű feladatoktól kezdve egészen a komplex, összetett alkalmazások fejlesztéséig jól használható. Egyik nagy előnye hogy platform független, így bármilyen operációs rendszeren (Windows, unixok, Linux, MacOS) futtatható az nyelv által létrehozott alkalmazás. A programnyelv egyik legjelentősebb kiterjesztése a Tk (toolkit), amelynek a segítéségével különféle grafikus felhasználói felületeket (Graphical User Interface – GUI) hozhatunk létre a felhasználók számára. Az Ulyxes-nek az egyik elemi részét lépező TCL API is ezen a programnyelven íródott. [BBWK]

PHP: A PHP (Hyperterxt Preprocessor) alapvetően egy kiszolgáló oldali programozási nyelv. Általában HTML oldalakon használják, valamint fontos jellemzője, hogy a program a szerveren illetve szervergépen kell, hogy fusson. A hagyományos HTML lapokkal szemben a szerver a PHP parancsokat nem küldi ki a felhasználónak, hanem a kiszolgáló oldalon egy értelmező dolgozza fel azokat. A PHP szkirptek használatával adatbázis lekérdezéseket végezhetünk el, különböző fájlokat olvashatunk és írhatunk és végül, de nem utolsó sorban távoli kiszolgálókkal létesíthetünk kapcsolatot. AZ Ulyxes a PHP segítéségével kérdezi le az téradatokat egy adatbázis szerverről (PostGIS) majd továbbítja az adatokat az Apache web szerver felé.[MAZA]

PostgreSQL/PostGIS: A PostgreSQL egy platform független, objektum-relációs adatbázis kezelő rendszer (Object Relational Database Management System – ORDBMS), mely hozzájárul a különböző adatok strukturált tárolásához, valamint az adatbázis működéséhez. A PostGIS viszont hozzá egy bővítmény, melynek segítségével térinformatikai adatokat tárolhatunk az adatbázisban. A PostGIS a rendszeren belül a különböző szenzorokból származó adatok tárolására szolgál.

MapServer: Ez egy olyan nyílt forráskódú, platform független projekt, mely a dinamikus térbeli adatok alapján meghatározott térképek interneten történő megjelenítésért felel. A Mapserver egy igen sokoldalú megvalósítás, mely szinte az összes raszter, vektor és adatbázis formátumot támogatja, ezzel számos lehetőséget kínálva a felhasználó számára.

5.2. Az rendszer részei

5.2.1. TclAPI

Szakdolgozat készítése során lehetőségem nyílt arra, hogy megismerkedjek az Ulyxes rendszer meghatározó részét képező TclAPI-val. A TclAPI egy tcl programnyelvben íródott, magas szintű alkalmazás-programozói felület. Használatával a felhasználónak lehetősége nyílik viszonylag alacsony programozói tudással, vezérlési műveleteket és automatizált mérések végrehajtására a különböző típusú robot mérőállomásokon. Az interfész jelentősége még abban is rejlik, hogy elrejti a különbségeket az egyes mérőállomások között, ezzel viszonylag nagyfokú rugalmasságot biztosítva.

Az API használatbavétele előtt szükség van a Tcl programnyelv 8.3-as vagy ennél újabb verzióját telepíteni a munka során felhasznált számítógépekre. Másfelől pedig az API működéséhez szükség van soros vonal bemenetre a számítógépen, ennek hiányában azonban USB-soros vonal átalakító használatával is megoldható a műszer-számítógép közötti kommunikációs kapcsolat. Ami az API telepítését illeti, letöltést követően csak egyszerűen ki kell csomagolni az Ulyxes oldaláról (http://www.agt.bme.hu/ulyxes/index_hu.html) letöltött fájlt, majd ezt követően a TclAPI használatra kész. Természetesen ahhoz, hogy tesztelni és használni lehessen az imént említett interfészt, olyan mérőállomásra lesz szükségünk, melynek használata és kapcsolata az API-val már implementálva lett, esetlegesen tesztelve is lehet. A TclAPI eddig gyakorlatilag 3 fajta robot mérőállomással lett letesztelve, melyek a következők:

- Leica TCA1800
- Leica 120x
- Trimble 550x

A gyakorlati alkalmazásiam során a Leica típusú robot mérőállomásokat alkalmaztam, valamint az egyes javításokat és újabb programok tesztelését is ezeken végeztem el.

A TclApi következő alapvető forrás (source) fájlokkal rendelkezik:

- *calc.tcl*: A calc.tcl különféle számítások valamint lista kezelések elvégzésére szolgáló algoritmusokat tartalmazza, melyek függetlenek a mérőállomások típusaitól. Ilyenek például az egyes szögátváltások végrehajtására szolgáló függvények, avagy egy koordinátalista betöltése a memóriába;
- *global.tcl*: Ezen rövid szkript fájl a többi programrészben felhasznált globális változókat tartalmazza. Pl.: a π pontos értéke;
- *tca1800.tcl*: A tca1800.tcl nevű fájl teszi lehetővé a Leica műszerek vezérlését. A program kiegészítőjeként szolgál a tca1800.com, mely az egyes soros vonallal kapcsolatos paraméterek listáját tartalmazza;
- *trimble.tcl*: Ennek a segítségével pedig a Trimble 550x típusú műszerek vezérlése válik lehetővé a számítógépen keresztül.

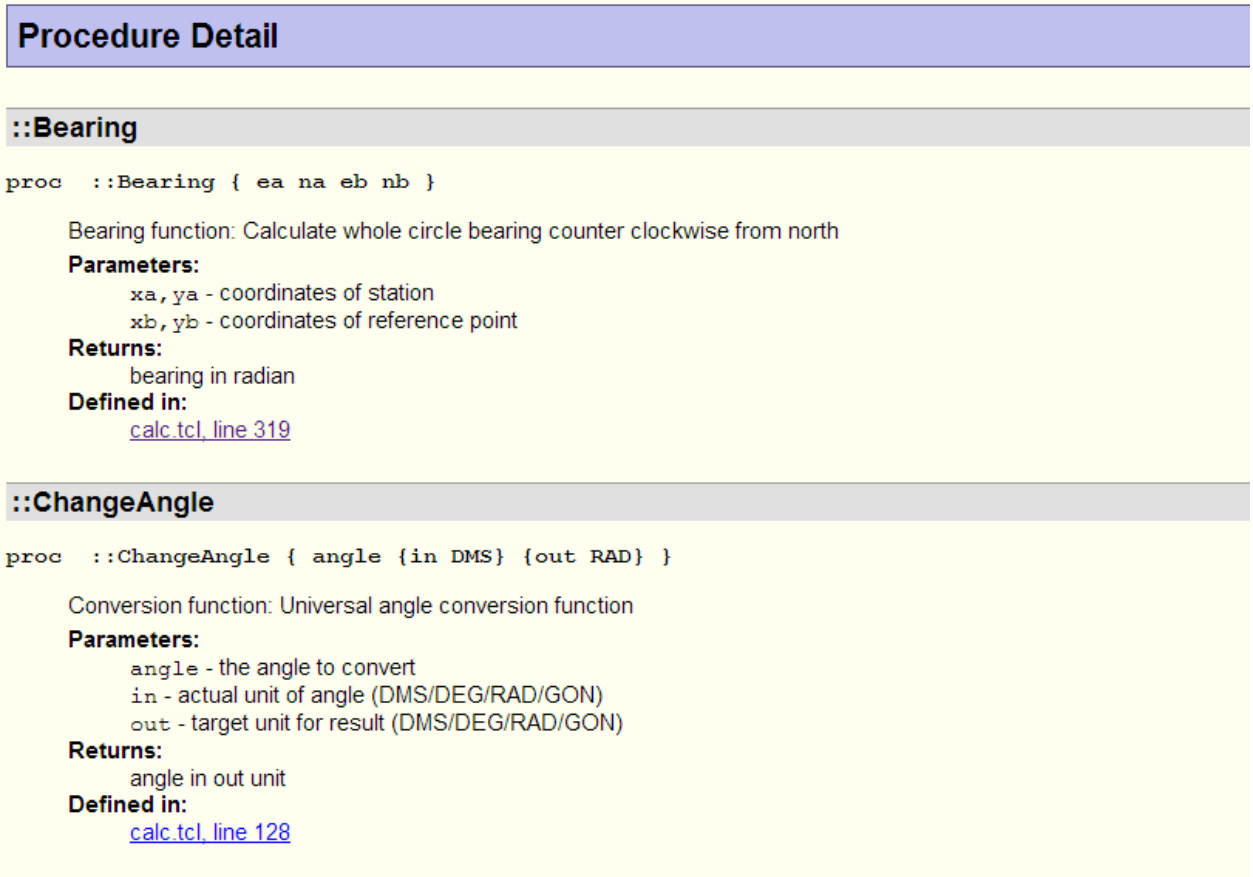
Az itt említett fájlokon kívül számos egyéb bár a forrásfájlokra épülő alkalmazást (szkriptet) tartalmaz a TclAPI, melyek felhasználásával különböző feladatok (pl.: homlokzatmérés) végrehajtását automatizálhatjuk és kezelhetjük. Ahhoz hogy a TclAPI-ban lévő függvényeket, rutinokat az egyes felhasználók könnyen áttekinthető és strukturált formában lássák, egy fejlesztői dokumentációt készítettem, melynek neve: **Ulyxes TclAPI documentation**. Felhasználói dokumentációra akkor van szükség, amikor az adott programot (mi esetünkben alkalmazás-programozói felületet) a készítőn kívül más felhasználók is használni fogják. Számukra biztosítani kell egy olyan, interneten elérhető dokumentációt illetve leírást, melynek használatával egyértelműen világossá válik a programban lévő metódusok működése valamint a program tevékenységi köre.

A „TclAPI documentation” készítése során a Tcldoc névre hallgató dokumentáció generátort használtam, mely képes HTML formátumú API dokumentációt készíteni az tcl forrás kódokból. Annak érdekében, hogy ezt a műveletet sikeresen végre tudjuk hajtani, különböző annotációkkal (megjegyzések) kell ellátni a programot, természetesen a megfelelő szintaxissal. Az annotációk azt a célt szolgálják, hogy egy adott programot vagy procedúrát (függvényt) elássuk különféle megjegyzéssel. A TclAPI dokumentáció készítésénél a következő annotációkat használtam fel:

- @author: megadja az adott program illetve procedúra szerzőjének a nevét;
- @version: az adott program verzió számát megadó annotáció;
- @param: egy procedúra paramétereit megadó annotáció;
- @return: egy adott procedúra visszatérési értékét jellemző annotáció.

Az elkészített dokumentáció több részből tevődik össze:

- Overview: ezen áttekintő rész összefoglalja a TclAPI feladatkörét, leírja az egyes fejlesztők nevét, valamint az API követelményeit részletezi;
- Procedure Summary: egy adott szkripten belüli procedúrák nevét, bemeneti paramétereinek a megnevezését valamint a feladatait ismerteti;
- Procedure Detail: A Procedure Detail viszont már részletezi az egyes procedúrák bemeneti paramétereit, visszatérési értékeit, valamint egy hivatkozást találhatunk arról, hogy az adott szkriptfájl-ban melyik sorban található.



5-1.ábra: A Bearing és a ChangeAngle függvény részletes leírást tartalmazó Procedure Detail

A TclAPI tehát a robot mérőállomások vezérlését teszi lehetővé. Annak érdekében, hogy az egyes szenzorokkal automatizált méréssorozatokat lehessen végrehajtani, saját fejlesztésű alkalmazásokat kell írni, persze csak abban az esetben, ha az adott feladat megoldását még nem készítette el valaki. Az egyes alkalmazások létrehozását célszerű az API programnyelvével, a tcl segítségével létrehozni, mely esetekben a TclAPI függvényeit célszerű felhasználni. Bár egy kicsit összetettebb automatizált feladat végrehajtásához mindenképp szükség van legalább egy pár soros kis alkalmazást írni, azonban az API egyes függvényei felhasználói szintű (High level)

metódusok is, melyekkel például egy mérést, vagy egy műszertől való koordinátalekérdezést könnyedén végrehajthatunk.

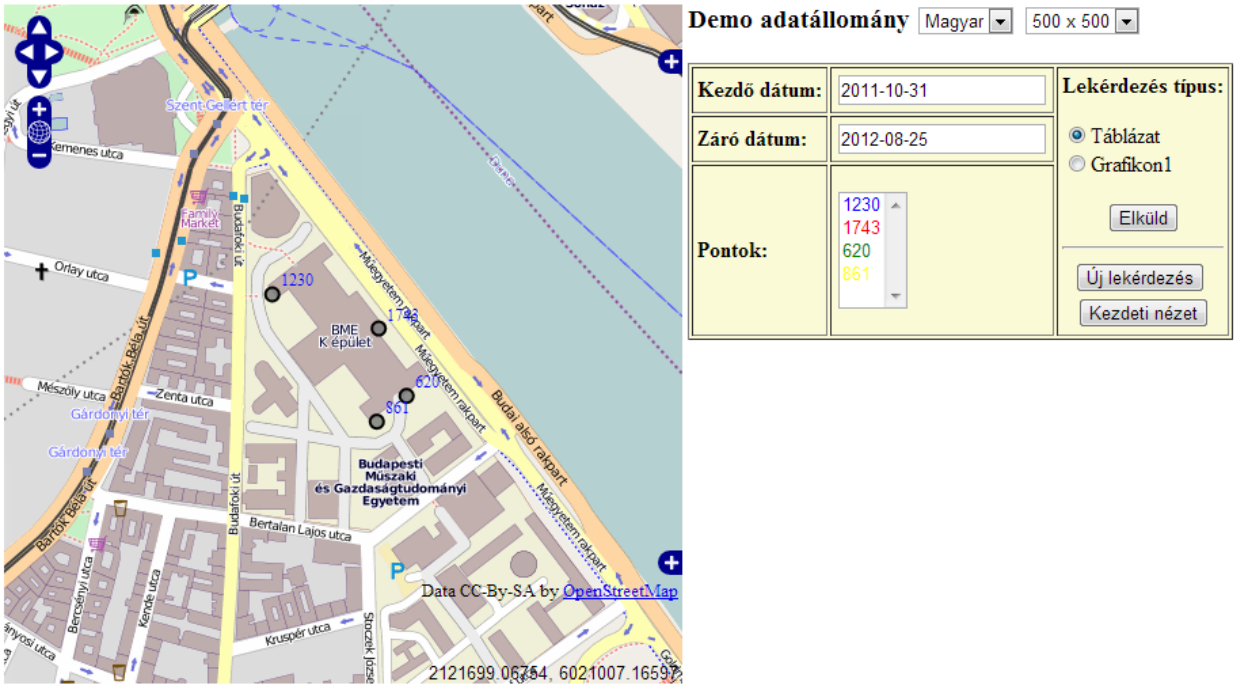
Az API felhasználói szintű függvényei mellett, található számos hardver szintű (Low level) eljárás is, melyek a műszer és számítógép kommunikációjával kapcsolatos feladatokat hatják végre. Annak érdekében, hogy egy egyszerű felhasználói szintű parancsot végrehajtsunk a műszeren, néhány függvényhívásnak eleget kell tennünk. Az API használatbavételéhez elsőként is mindenképp szükség van az TclAPI valamint a tcl programnyelv meglétére a felhasználó számítógépen.

Egy felhasználó szintű metódus végrehajtásához a következő sorokat kell beírni a parancssori ablakban:

- tclsh: Ez a parancs a tcl programnyelv fordítójának felel meg. A tclsh beírása után a fordító soronként beolvassa a parancsokat az ablakból, majd lefordítja és végrehatja őket, kiírva az egyes eredményeket illetve hibaüzeneteket;
- source *global.tcl* – Elsőként is be kell importálni a globális változóinkat, melyek az egyes függvények végrehajtásának az elengedhetetlen részét képezik;
- source *calc.tcl* – Ez a sor felel azért, hogy az egyes számítási műveleteket tartalmazó algoritmusokat használatba tudjuk venni;
- source *tca1800.tcl* – Ennél a parancsnál az a mérés során használni kívánt műszerhez tartozó vezérlő programját kell beírni. Abban az esetben, ha a Leica típusú műszereket szeretnénk alkalmazni, akkor *tca1800.tcl* nevű alkalmazást kell meghívni, ha pedig a Trimble műszerekkel szeretnénk foglalkozni akkor a *trimble.tcl*-t kell beírni a source parancs után;
- Ezt követően pedig különböző vezérlési műveletet hajthatunk végre a műszereken, például:
    - Move *hz v*: Paramétereként megadott irányba fordítás
    - Measure: Távmérés végézése és eredményének a visszaadása
    - Coords: koorditának lekérdezése a műszerből.

5.2.2. Megjelenítést lehetővé tevő felület

Az Ulyxes rendszer második részét a megjelenítő felület képzi, melynek használatával a mérési eredményeinket és az ehhez kapcsolódó elemzéseinket publikálhatjuk táblázatok és grafikonok formájában az ezekhez tartozó térinformatikai megjelenítéssel együtt. A felület működtetéséhez a felhasználónak szüksége van internetkapcsolatra, valamint egy böngészőre is, mely támogatja a javascript-ek futtatását. Operációs rendszer szempontjából platform független az alkalmazás. A térképi megjelenítéshez használt térinformatikai rendszer hátterét az Open Street Map (OSM) szolgáltatja. Az OSM egy bárki számára szabadon felhasználható utcatérkép, mely az interneten megtalálható. Az eredmények publikálásához, grafikonok megjelenítéséhez szükséges adatok tárolását viszont egy web szerver teszi lehetővé, melynek pontos részletezése a következő alfejezetben található.



5-2. ábra Adatok publikálást lehetővé tevő felület OSM alkalmazásával

5.2.3. Szerverek

A rendszer rugalmas működéséhez számos nyílt forráskódú projekt illetve szerver hozzájárul, melyek segítségével az adatok tárolása, áramlása és megjelenítése biztosított. Annak érdekében, hogy különböző szenzorokból származó adatokat egy adatbázisba kerüljenek, a PostsgreSQL adatbázis kezelő rendszer PostGIS kiegészítőjét használja fel az Ulyxes. A PostGIS kiegészítő jelentősége abban rejlik, hogy segítségével a munkák során bemért téradatok tárolása

válik lehetővé az adatbázisokban, melyek a PostGIS szerverén helyezkednek el. Amint már említésre került, az adatbázisból az adatok áramlását a PHP programnyelv segítségével valósítja meg a rendszer a web szerver felé. Az Ulyxes projekt során felhasznált web szervert a szintén szabadon használható Apache Web Server szolgáltatja. Az itt említett szerverek használata csak egy alternatíva az Ulyxeshez kapcsolódóan. Ezeken kívül még kialakítható olyan megjelenítési módszer is mely a Google Maps szerver térinformatikai adatait használja a térképi megjelenítéshez, valamint olyan lehetőség is létezik, melynél egy Map Server (pl.: WMS) biztosítja az adatok áramlását az adatbázis szerveréről a web szerver felé, azonban ezek a megvalósítási módok még nem készültek el teljes körűen a z Ulyxes-et illetően.

5.3. Gyakorlati alkalmazások a rendszer felhasználásával

A gyakorlati alkalmazások kivitelezésénél az volt a feladatom és egyben a célom is, hogy bemutassam a robot mérőállomások használatát, eltérő mozgásvizsgálati munkák során, érzékeltetve az egyes módszerek előnyeit és hátrányait összevetve más geodéziai technikákkal.

5.3.1. Gyakorlati alkalmazások során felhasznált szoftverkomponensek és segédeszközök

Az adatok feldolgozását nyílt forráskódú programok segítségével végeztem el, melyek a következők voltak:

- Octave : Az Octave egy magas szintű interaktív környezetet lehetővé tevő programozási nyelv, melynek használatával különféle numerikus számításokat, vizuális megjelenítéseket és programozási megoldásokat vihetünk véghez. Használatával a felhasználó adatok sokaságán végezhet elemzéseket, algoritmusokat fejleszthet és végül, de nem utolsó sorban alkalmazások készítésére is tökéletesen megfelel. Az Octave a sokak által ismert Matlab programnyelv nyílt forráskódú változata, ingyenesen letölthető az internetről, valamint a forráskódjához bárki szabadon hozzáférhet, esetleg módosíthat rajta. Ezzel a lehetőséggel bárki igénye szerint változtathatja a szoftver működését, esetenként tovább is fejlesztheti azt.
- Libre Office – Calc: A Libre Office programcsaládnak az egyik legtöbb lehetőset tartalmazó része a Calc nevű program. Bár a piacon, hazánkban ez a második legelterjedtebb táblázatkezelő (a Microsoft Office – Excel után), azonban ha ingyenes programok között keressük a legjobbat, az első helyen áll. Alkalmazásával egyszerűbb számításokat, grafikonokat, adatdiagramokat készíthetünk, melyek egy adott munka során hasznunkra válhat.

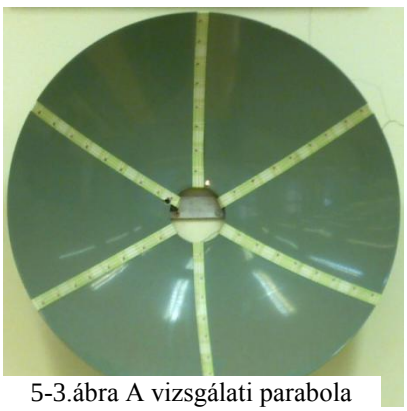
- GeoEasy: A GeoEasy a DigiKom Kft. által fejlesztett geodéziai programcsomag, mely mérőállomásokkal rögzített adatállományok feldolgozására szolgál. A program segítségével alappontok és részletpontok koordinátáinak a számítását végezhetjük el, beleértve ezen koordináták és az ezekkel kapcsolatos jegyzőkönyvek megjelenítését, illetve nyomtatását. A program nagy előnye, hogy Linux és Windows operációs rendszerek alatt egyaránt alkalmazható, ezzel támogatva a különböző igényű földmérők munkáját.
- Structorizer: A Struktorizer egy Luxemburgi csapat által készített, ingyenesen használható szoftver, melynek alkalmazásával egyszerűen és elegánsan tudunk készíteni különböző struktogrammokat. Ezen struktogrammokkal könnyedén és érthető módon lehet szemléltetni egy adott program algoritmusát. A szakdolgozatom során készített programok közül csak azokat láttam el ilyen ábrákkal, melyeknek terjedelme megengedte ezt, ugyanis nagyobb terjedelmű programok leírására ezen módszer nem alkalmazható.

5.3.2. Parabola mérése: *scan.tcl* program bemutatása

A feladat elvégzése során a célom az volt, hogy megvizsgáljam a robot mérőállomások hatékonyságát a szkennelési feladatok elvégzését illetően, összevetve a manuális, kézzel történő mérésfolyamat valamint a lézerszkenner munkaképességével. A mérés elvégzéséhez rendelkezésre állt a *scan.tcl* nevű program, mely TclAPI használatának a segítségével valósul meg. A program közvetlenül nem rendelkezik bemeneti paraméterrel, azonban a program működéséhez társul egy *scan.par* paraméter lista is, melyek a következő, mérés beállításával kapcsolatos paramétereket tartalmazza:

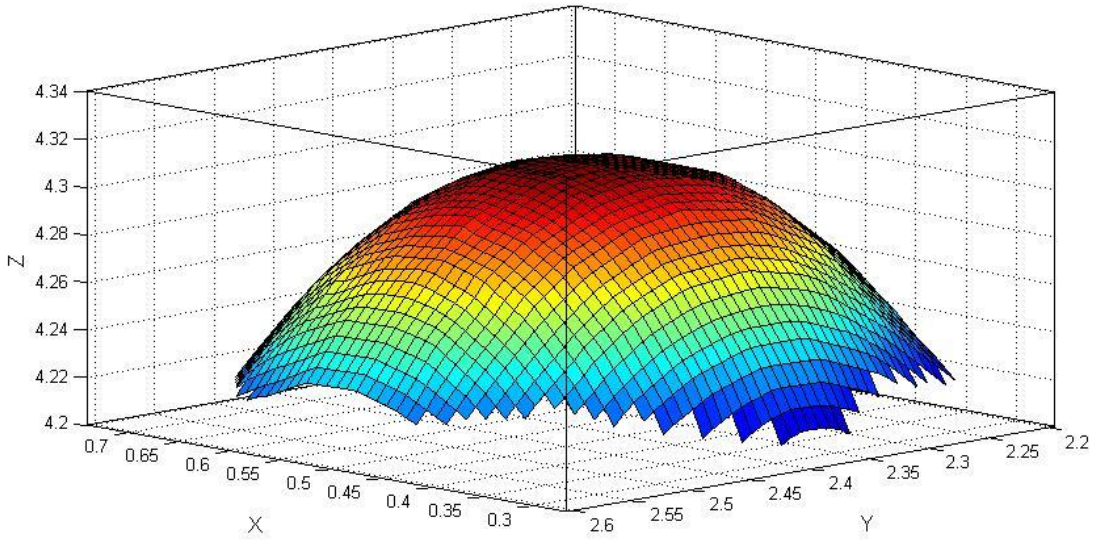
- step: a szkennelési folyamat lépésközének az értékét beállító számérték;
- hz_min, hz_max: a mérési tartomány minimális illetve maximális vízszintes szögét megadó paraméter;
- v_min, v_max: a mérési tartomány minimális illetve maximális magassági szögét megadó számérték;
- compar: a műszer-számítógép közti kapcsolat kommunikációs csatornáját beállító paraméter.

A *scan.tcl* tesztelését a BME Általános és Felsőgeodéziai tanszék komparátor termében elhelyezkedő parabolán hajtottam végre. A feladat során a Leica 1201-es robot mérőállomást



5-3.ábra A vizsgálati parabola

használtam. Ami a szkennelés gyorsaságát illeti, percenként 12 pontot tudott lemérni a műszer, ami öt másodpercenként egy pont bemérését jelenti. A mérés elvégzését követően, egy koordinátafájl állt a rendelkezésemre, melynek a feldolgozását és elemzését az Octave és a GeoEasy segítségével végeztem el. Elsőként is egy adatszűrést kellett elvégeznem az adatsoron, melynek során a parabolán kívül eső pontokat kitöröltem, majd ezt követően a GeoEasy segítségével egy gömböt illesztettem a parabolára. A regressziós gömb a következő paraméterekkel rendelkezett: Y_0 = 2.388, X_0 = 0.503, Z_0 = 4.005, R = 0.309. Az regressziós feladat sikeresnek mondható, mivel az illesztés pontosságát megadó négyzetes átlagnak (Root Mean Square – RMS) nevezett számérték 0.002 m-re, azaz 2 milliméterre adódott. Az illesztést követően az Octave segítségével lineáris interpoláció használatával ábrázoltam a mért parabolát. Az így kapott eredményt a következő kép ábrázolja:



5-4. ábra A mért parabola ábrázolása lineáris interpoláció segítségével

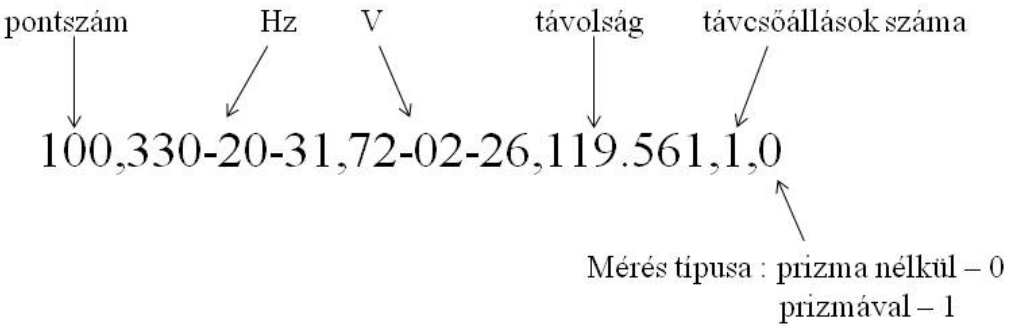
A mérés eredményeit tekintve elmondható, hogy módszer hatékonysága a manuális mérésnél számottevőbben jobb. Az szkennelés automatizálásával szignifikáns mennyiségű időt takarít meg az ember, mivel csak a kezdeti paramétereket kell beállítani a mérést megelőzően. Az itt használt metódust egy lézerszkenner feltehető produktivitásával összehasonlítva az elmondható, hogy a mérőállomások nem veszik fel a versenyt, azonban kisebb feladatoknál, mint a példában említett, megfelelő alternatívája lehet.

5.3.3. Z épület deformáció mérése

A feladat célja a BME Z épületének hőmérséklet hatására bekövetkező deformációjának az észlelése. A feladat végrehatása során a Leica 1201-es műszert alkalmaztam, mivel a két rendelkezésre álló, TclAPI-val kompatibilis műszerek közül csak ez rendelkezik reflektor nélküli távmérővel. A méréseket olyan időpontokra időzítettem, amikor a nap süti a Z épület déli oldalát, mivel ezekben az időpontokban biztosan feltételezhetjük azt, hogy a déli oldal számottevően nagyobb deformációt „szenved” mint az északi oldal.

Ahhoz hogy a feladatnál a legnagyobb pontosságot érjük el, prizmák felszerelésére lett volna szükség az épületre. Sajnos erre nem volt lehetőség, ezért a *scan.tcl* meglévő program segítségével próbáltam az épület déli oldalait leméri. Itt azt tapasztaltam, hogy az automatizált program végrehajtása során a mérés többször sikertelen eredményhez vezetett, mivel sok olyan pont esett a *scan.tcl* program vizsgálati tartományába, melyeknél a reflektor nélküli távmérő nem tudott mérni (pl.: ablak). Ezt követően arra a megoldásra jutottam, hogy készíteni kell egy vízszintes és magassági szögeket tartalmazó pontlistát, melyet paraméterként a *robot.tcl* névre hallgató programnak megadhatunk. A *robot.tcl* gyakorlatilag egy mérés automatizáló alkalmazás, melynek segítségével a paraméterként megadott pontlistát, a megadott sorrendben leméri. A pontlistába a mérési szögeken kívül olyan értékeket is tartalmaz, melyekkel meghatározhatjuk, hogy reflektor nélküli vagy prizmás távmérőt akarunk e használni, valamint a távcsőállások számát megadó paraméter is tartalmazza. A *robot.tcl* eredményfájlnak a vége tartalmazza a mért pontok koordinátáit.

Annak érdekében, hogy a mérési eredményeket tartalmazó pontlista elkészítését szintén automatizálni lehessen, készítettem egy *scanfileMaker.tcl* nevű programot, melynek használatával a *robot.tcl*-nek paraméternek beadható formátumú (tca) mérési pontlistát lehet készíteni. A *scanfileMaker.tcl* program által kiadott output fájl egy sorának a magyarázatát a következő ábra szemlélteti:



A munka végrehajtásánál több mérési adatsor keletkezett, melyek közül hármat választottam ki:

- Első mérés: Időpont: 2012.09.26 $T_1 = 24°$ Páratartalom = 54%
- Második mérés: Időpont: 2012.10.4 $T_2 = 14°$ Páratartalom = 65%
- Harmadik mérés: Időpont: 2012.10.30 $T_3 = 4°$ Páratartalom = 60%

Az automatizált mérések elvégzése előtt a tájékozást a Buda Vár kupolájára végeztem el, valamint a tájékozás pontosságának az érdekében minden mérésnél a Kálvin és a Bakáts toronyra hajtottam végre az ellenőrző méréseket.

5-5. ábra Z épület vizsgálati pontjai

Az automatizált mérés során, az épületen 37 darab, adatsoronként ugyanabba az irányba eső pontok (5-2-es ábrán sárga színnel jelölt pontok) koordinátája lett lemérve. Az mérések eredményeit GeoEasy és Libre Office program segítségével dolgoztam fel. Elsőként is regressziós számításokat [JFOX] hajtottam végre az egyes adatsorokon, melyeknek a végrehajtása során az koordináta halmazokra Z koordináta értékben változó, Y-X kiegyenlítő síkot illesztettem a GeoEasy program segítségével. A síkok illesztésének az eredményeképpen rendelkezésemre állt, három darab sík, melyek által bezárt szögek megegyeznek a normálisuk által bezárt szögével. A Z épület hőmérséklettani mozgásával kapcsolatban azzal a feltételezéssel éltünk, hogy napos időben az épület alja viszonylag mozdulatlan marad, a teteje viszont az aljához képest észak felé elmozdul. Ezt követően a regressziós síkok egyenletéből adódó normális vektorok skaláris szorzatának az egyenletéből a síkok által bezárt szögek a következő értékek lettek:

- Az első és a második mérés között: 0-0-5,0
- A második és harmadik mérés között: 0-0-10,8
- Az első és a harmadik mérés között: 0-0-12,6

Az itt kapott eredményeket vizsgálva arra lehet következtetni, hogy az egyes regressziós síkok a hőmérséklet csökkenés hatására azonos irányba mozognak. Az eredmények pontosabb megvizsgálása érdekében szükség volt az Office programba importálni az egyes mérésekhez tartozó koordinátákat. Ennél az elemzési résznél szintén azt a trendet lehetet megfigyelni, hogy az épület tetején lévő pontok hőmérséklet hatására történő elmozdulásai nagyobbak voltak, mint az épület alsóbb részeire eső pontokénak. Az első (T=24°) és a harmadik (T=4°) mérés esetén a legfelső 5 pont térbeli elmozdulás vektorának az átlaga 5,5 mm-re adódott.

A feladat folyamatát tekintve, a robot mérőállomások segítségével könnyen abszolválható a feladat végrehajtása. A TclAPI-ra támaszkodó *robot.tcl* valamint az újonnan készített *scanfileMaker.tcl* segítségével szinte teljes mértékben automatizálni lehetet a mérés menetét, csak a műszer tájékozásának valamint a meteorológiai adatok megadásának kell eleget tenni minden mérés kezdeténél.

5.3.4. Szabadsághíd mozgásának vizsgálata a rajta áthaladó járművek hatására

A feladat célja az, hogy meghatározzam a Gellért heggyel szemben elhelyezkedő Szabadság híd lehajlását és vízszintes mozgását a rajta áthaladó forgalom hatására valamint kimutatni azt, hogy van e valamilyen összefüggés a híd két oldalán mért adatsorok között. A

5-6. ábra Szabadság hídon környékén elhelyezkedő prizmák és műszerek felülnézetben

feladat végrehajtása során három darab, különböző típusú mérőműszert használtunk: két darab TCA1800-as és egy darab TPS1201-es típusú Leica robot mérőállomást. Az utóbbira azért volt szükség, hogy a híd egyik pillérére távolságméréseket tudjunk végezni a prizma nélküli távmérő segítségével.

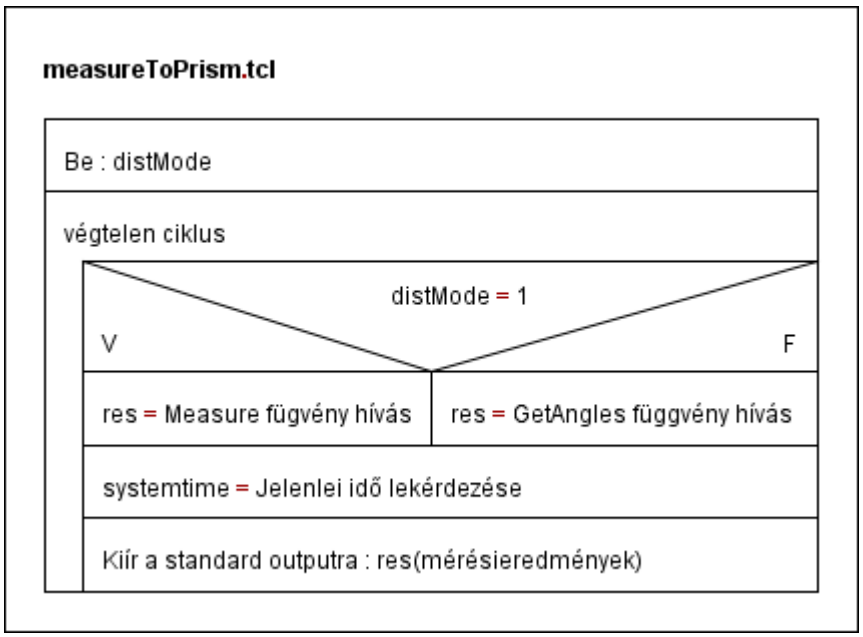
A feladat kivitelezésével kapcsolatos előkészületeket a komparátor teremben végeztük el. Elsőként a szükséges vezérlő programokat készítettem el a Tcl programnyelvben, melyek a következők:

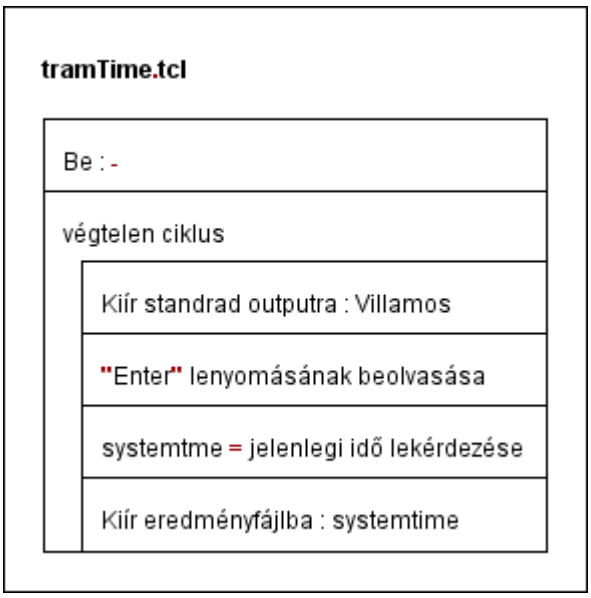
- measureToPrism.tcl – A measureToPrism nevű program lehetővé teszi azt, hogy egy adott pontra (prizmára) folyamatos és automatikus méréseket lehessen végezni, feljegyezve minden egyes méréshez a pontos időpontot és mérési eredményt a standard output-ra. A program rendelkezik két darab parancssori paraméterrel is, melyek közül az

elsőnek köszönhetően a felhasználó opcionálisan eldöntheti, hogy szeretne-e távolságmérést végezni, a második paraméternek viszont a műszer típusára vonatkozó kommunikációs csatornát lehet beállítani.

- tramTime.tcl – Ennek a programnak köszönhetően, a híd mindkét oldalán fel tudtuk jegyezni, azokat az időpontokat, melyeknél a villamosok a híd közepén helyezkedtek el, ezeket felhasználva a későbbi feldolgozásokhoz.

A két programnak a struktogrammját a következő két ábra prezentálja:





Az előkészületi munkák során világossá vált, hogy a méréshez két darab számítógépre (laptop) lesz szükségünk, mivel a műszerek közötti távolság és a keresztező forgalom nem tette lehetővé a műszerek az egyetlen gépről történő vezérlését. A mérés megkezdése előtt a két számítógépen időszinkronizációt hajtottunk végre. Az időszinkronizáció segítségével különböző szerverektől kérdezhetjük le a pontos időt, ami a mi esetünkben lényegében azt jelentette, hogy a két gépen lévő idő azonos értéket mutassanak, másodpercre pontosan. Ennek a megvalósítását a Windows rendszer internetes beállításainál tudtuk végrehajtani. A szinkronizálást során time.windows.com internetes kiszolgálóval végeztük el.

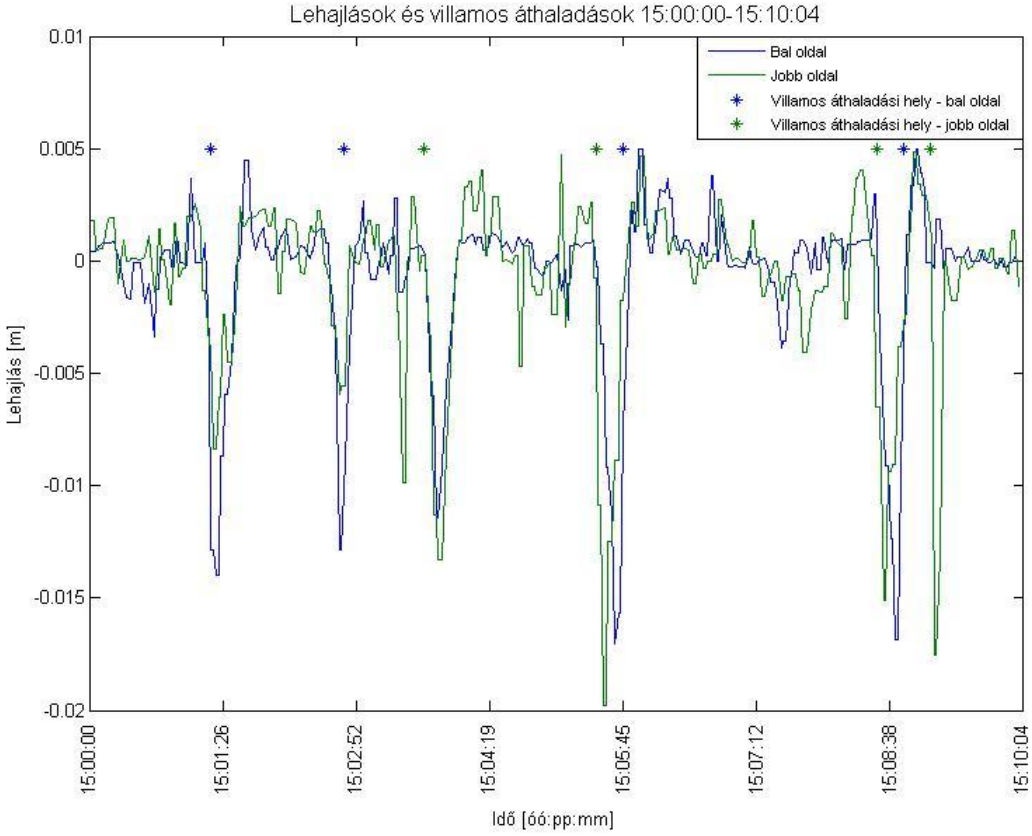
A mérés elvégzését egy délutáni napra időzítettük, mivel ekkor közlekednek a villamosok a legsűrűbben, melyek várhatóan a legnagyobb deformációt okozzák a híd szerkezetében. A munka első lépésében felálltunk a műszerekkel a Szabadság híd elején elhelyezkedő kőpárkányokra Kern pillértalpak segítségével. A híd közepén mindkét oldalon

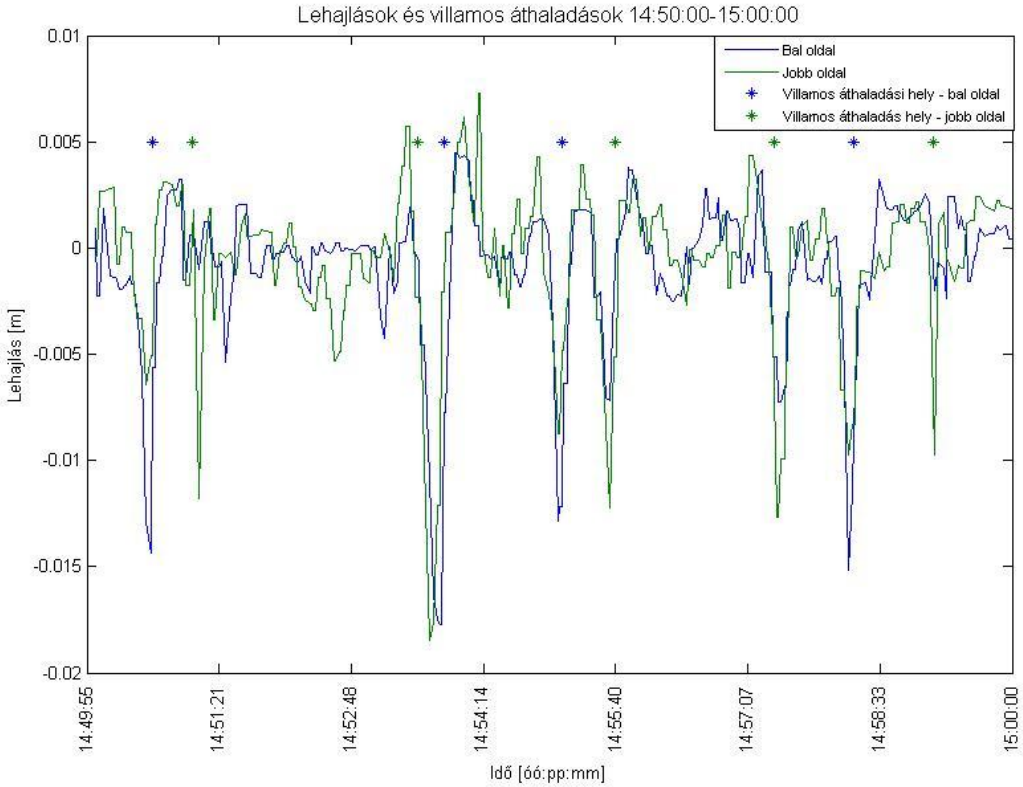
pillanatszorítókkal rögzítettük a prizmákat, melyekre a mozgásvizsgálati méréseket terveztük végezni. Ezt követően pedig elsőként beállítottuk minden egyes műszernél a megfelelő összeadó és meteorológiai állandókat. Egy helyi koordinátarendszert vettünk fel, melyet a két TCA1800-as műszer álláspontja határozott meg. Az vízszintes koordinátarendszer y tengelye jó közelítéssel, a híd hossztengelyével párhuzamos, a jobb oldali hídfőn elhelyezett műszer pedig a koordinátarendszer kezdőpontját adta meg. A műszerek munka közbeni elhelyezkedését a jobb oldali kőpárkányon pedig a következő kép szemlélteti:

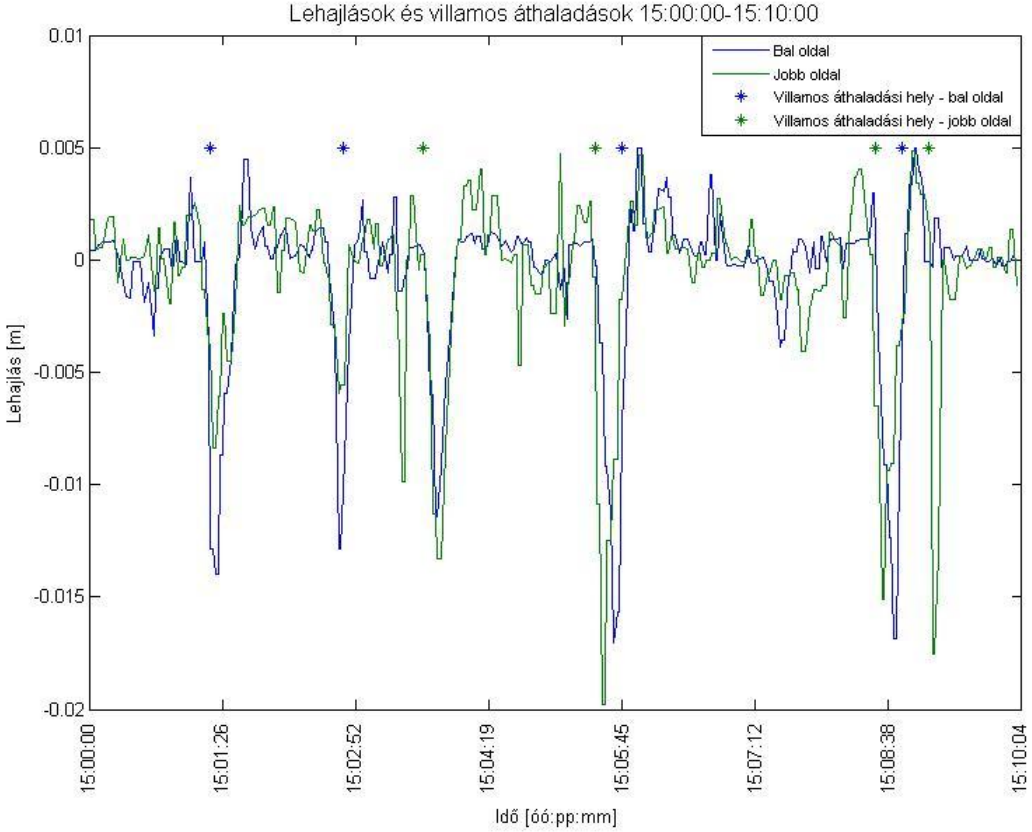
5-7. ábra Műszerek mérés közbeni elhelyezkedése - Jobb oldal

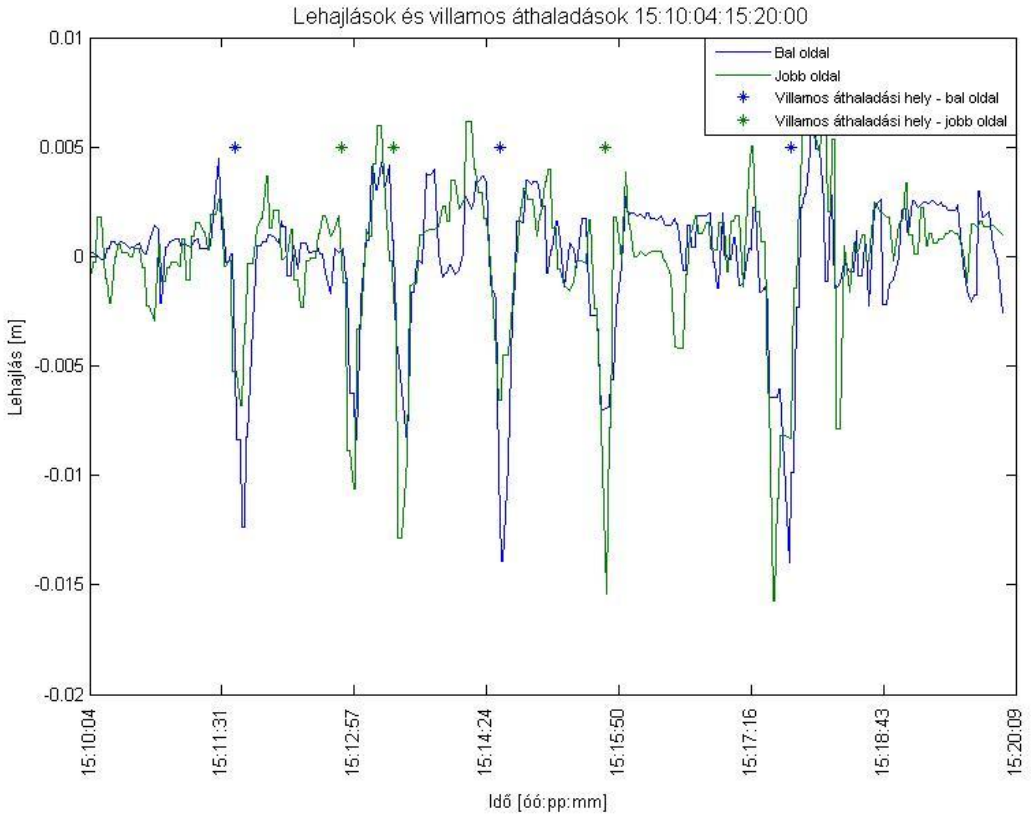
Az adatok feldolgozása során tehát az elsődleges cél a híd közepén bal és jobb oldalon lévő korlátok lehajlásainak, lehajlás változásainak kimutatása. A hídon végzett automatizált mérések eredményeképpen rendelkezésünkre állt egy több, mint 60 perces adatsor a létesítmény mindkét oldalára és a budai jobb oldali pilon dőléseire. A mérések kiértékelése előtt elengedhetetlen volt a két oldalon mért adatsorokat és a villamos áthaladások adatait egybevonni az időpontok alapján, a későbbi feldolgozások érdekében. Ennek elvégzéséhez készíteni kellett egy programot (*merge.tcl*), mely ezt automatikusan elvégzi. Ezt követően pedig a több, mint 1 órás, mérési eredményeket tartalmazó adatsorból ki kellett választani azokat a részeket, melyeknél méréseket külső zavaró tényezők nem befolyásolták. Ebből kifolyólag négy darab mérési adatsorra osztottam az eredményeket, majd ezeket az Octave programban ábrázoltam. Az egyes mérőállomások teljesítményeit vizsgálva viszont arra lehet következtetni, hogy a TCA1800-as műszerek 3-4 másodpercenként végeztek el egy mérést a prizmákra, ami óránként több mint 1000 mérésnek felel meg. Ami a TPS1201-es robotműszert illeti, 6-7 másodpercenként sikerült

egy helyzet-meghatározást végezni, azonban itt a mérés prizma nélküli módban történt. A adatsorok közöti lineáris kapcsolatot tekintve, nem mutatkozott kolleráció a két oldali mérések között, valamint a keresztkorrelációval kapcsolatban sem sikerült kimutatni kapcsolatot a két oldali lehajlás illete a pilon dőlése között.




Lehajlások és villamos áthaladások 14:50:00-15:00:00
Bal oldal
Jobb oldal
Villamos áthaladási hely - bal oldal
Villamos áthaladás hely - jobb oldal
Lehajlás [m]
0.01
0.005
0
-0.005
-0.01
-0.015
-0.02
14:49:55
14:51:21
14:52:48
14:54:14
14:55:40
14:57:07
14:58:33
15:00:00
Idő [óó:pp:mm]


Lehajlások és villamos áthaladások 15:00:00-15:10:00
Bal oldal
Jobb oldal
Villamos áthaladási hely - bal oldal
Villamos áthaladási hely - jobb oldal
Lehajlás [m]
0.01
0.005
0
-0.005
-0.01
-0.015
-0.02
15:00:00
15:01:26
15:02:52
15:04:19
15:05:45
15:07:12
15:08:38
15:10:04
Idő [óó:pp:mm]



Az adatok középhibáival kapcsolatban kiszámításra kerültek az Y,X koordináták kiszámító függvényeinek valamint a lehajlás kiszámítását lehetővé tevő függvényeknek a középhibái is a hibaterjedés képletével. Az egyes számértékek a következő képletekkel határozhatóak meg:

y = t_{ferde} *sin(Z) * sin(δ)

x = t_{ferde} *sin(Z) * cos(δ)

h = t_{ferde} * cos(Z)

A hibaterjedés képletét használva az egyes függvények középhibái a következőképpen alakultak:

$$my = \sqrt{\left(\frac{\partial y}{\partial\, \mathrm{t_{ferde}}}\right)^2 * m_{tf+}^2 \left(\frac{\partial y}{\partial \mathrm{z}}\right)^2 * \frac{m_z^2}{\rho^2} + \left(\frac{\partial y}{\partial \delta}\right)^2 * \frac{m_\delta^2}{\rho^2}} = 1mm$$

$$mx = \sqrt{\left(\frac{\partial x}{\partial\, \mathrm{t_{ferde}}}\right)^2 * m_{tf+}^2 \left(\frac{\partial x}{\partial \mathrm{z}}\right)^2 * \frac{m_z^2}{\rho^2} + \left(\frac{\partial x}{\partial \delta}\right)^2 * \frac{m_\delta^2}{\rho^2}} = 0{,}8mm$$

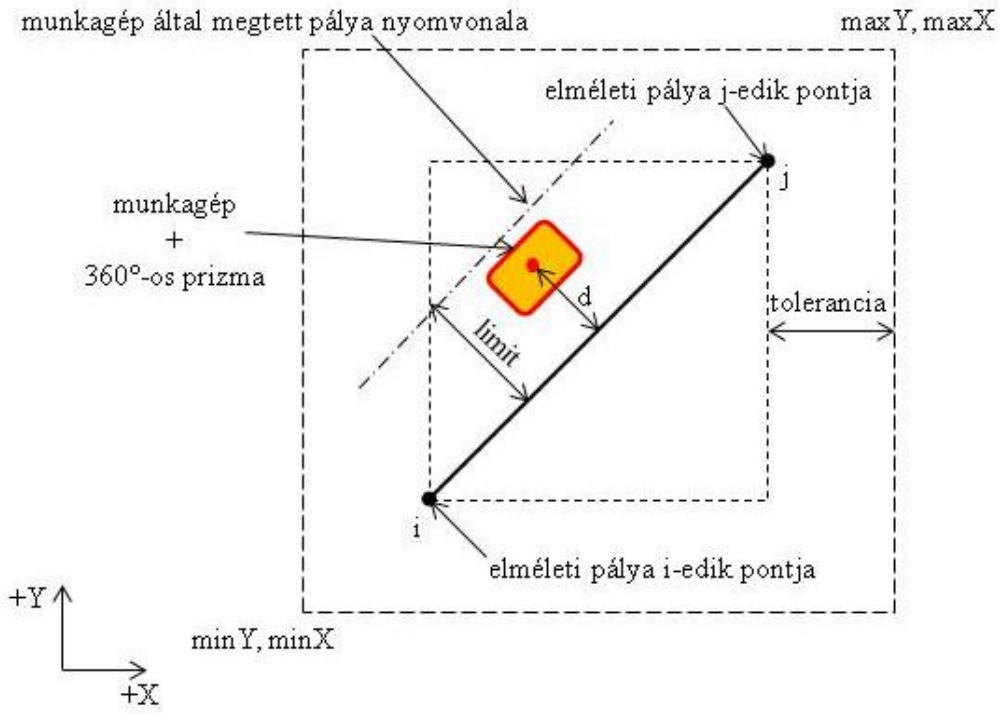
$$mh = \sqrt{\left(\frac{\partial h}{\partial\ \mathrm{t_{ferde}}}\right)^2 * m_{tf+}^2 \left(\frac{\partial h}{\partial \mathrm{Z}}\right)^2 * \frac{m_z^2}{\rho^2} + \left(\frac{\partial h}{\partial \delta}\right)^2 * \frac{m_\delta^2}{\rho^2}} = 0{,}8mm$$

Az itt kapott számértékeket, valamint a kiszámolt koordinátamozgások eredményét összevetve az állapítható meg, hogy szignifikáns mozgás semelyik koordinátatengely irányában nem volt megfigyelhető

5.3.5. „Munkagép" vezérlés

A korábbiakban ismertettek szerint a munkagépek vezérlése a GPS-es megvalósítás helyett robot mérőállomások használatával is kivitelezhető. A vezérlési feladat végrehajtása során a mi esetünkben a munkagépet egy személygépjárművel helyettesítettük, melynek a helyzetét folyamatosan vizsgáltuk vízszintes értelemben. A munka elkezdése előtt mindenképp meg kellett tervezni az irányító program struktúráját majd meg is kellett valósítani az alkalmazást. Az elkészült *measureToMachine.tcl* munkagép vezérlő program készítése során több átalakításon és tesztelésen ment keresztül, melyek mind a program teljesítményét és produktivitását javították.

A program koncepciója a következő: Három bemeneti paraméterrel rendelkezik, ezek közül az első paramétere a limit, melynek a megadásával meghatározhatjuk, hogy mennyivel térhet le a munkagép egy adott pályától. A második paraméter toleranciát megadó számérték, mely azt a program futását tekintve azt a célt szolgálja, hogy olyan esetekben, melynél a munkagép nagyon távol van a pályán, akkor a program fölösleges számításokat ne végezzen, ezzel javítva a hatékonyságát. A tolerancia határát megadva egy téglalap határ keletkezik a pálya körül, mely rendelkezik minimum és maximum Y, X (az x.-edik ábrán: minY, minX, maxY, maxY) koordinátákkal. Az utolsó, harmadik paraméter pedig egy koordináta lista, mely tartalmazza a követendő pálya vízszintes koordinátáit.



5-8. ábra MeasureToMachie.tcl működése ábrán szemléltetve

A program működését illetően azt fontos még megemlíteni, hogy a prizma és pálya vonalának a pillanatnyi távolságát egy adott pályaszakasz egyenesének az egyenletének a segítségével határozza meg. Az egyenes egyenletét a következő összefüggés adja meg:

$$a*x + b*y + c = 0$$

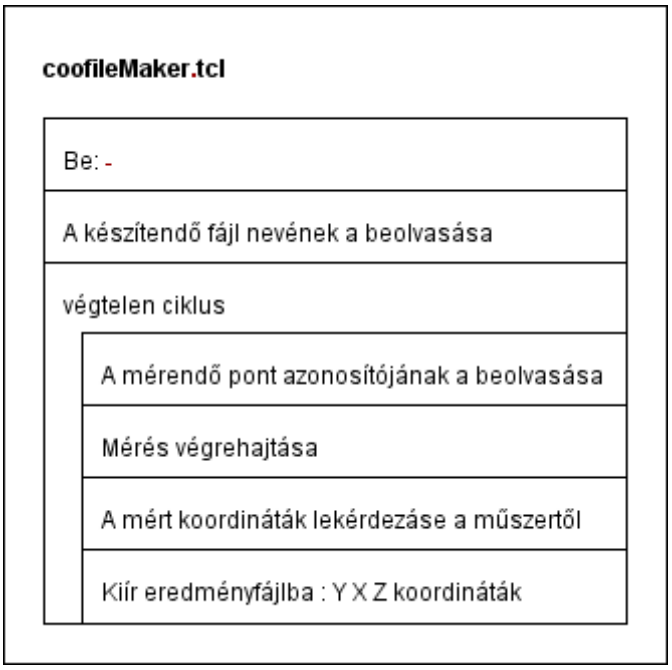
Ezt követően pedig a prizma és az elméleti pálya közti távolságot úgy kaphatjuk meg, hogy az előbbiekben leírt egyenes egyenletébe behelyettesítjük a vizsgált pont koordinátáit, majd osztjuk az x és y koordináta együtthatói által meghatározott vektor hosszával. Ennek a kiszámolása után a program összeveti a meghatározott távolságot a mi általunk megadott limit értékével. Ha a távolságunk átlépi a limitként megadott határértéket, akkor a program sípolással jelzi a számítógépen, hogy a jármű nincs az helyes úton. Ezek mellett a program egy *measureToMachine.log* nevű napló fájlba menti a „munkagép” helyzetét leíró koordinátákat, olyan megjegyzéssel kiegészítve, hogy rajta van e az adott pályán a megadott limiten belül vagy sem, természetesen a pályától való távolságot és az időpontot minden esetben feltüntetve.

```
The machine is on the right path on y: -2.9576 x: 1.9289 coordinates, distance : 0.7943m at 12:57:08

The machine is not on the right path on y: -3.8409 x: 0.9430 coordinates, distance : 1.0171m at 12:57:09
```

A feladat elvégzéséhez Leica 1201-es robot mérőállomást, valamint szintén egy Leica típusú 360°-os prizmát használtam fel. A prizma stabil elhelyezését a „munkagépre” egy mágneses adapter valamint egy kényszerközpontosító segítségével oldottuk meg. A munkagép irányításának a szimulációját a Savoya üzletház parkolójában valósítottuk meg. A munka megkezdése előtt szükség volt meghatároznunk az elméleti pályánkat. Ezzel kapcsolatban a kényelmes munka elvégzése érdekében készítésre került egy *coofileMaker.tcl* névre hallgató koordinátalista készítő alkalmazás. A program használhatóságát illetően elmondható, hogy bármilyen munkafolyamat során könnyedén felhasználható, előnye pedig, hogy az eredménylista formátumát és struktúráját tekintve, további átalakítás nélkül beolvasható a GeoEasy programba, ezzel a jövőbeli felhasználók számára is egy hatékony programot biztosítva. A *coofileMaker.tcl* struktogrammját a következő ábra szemlélteti:

5-9. ábra 360°-os prizma a "munkagépen"

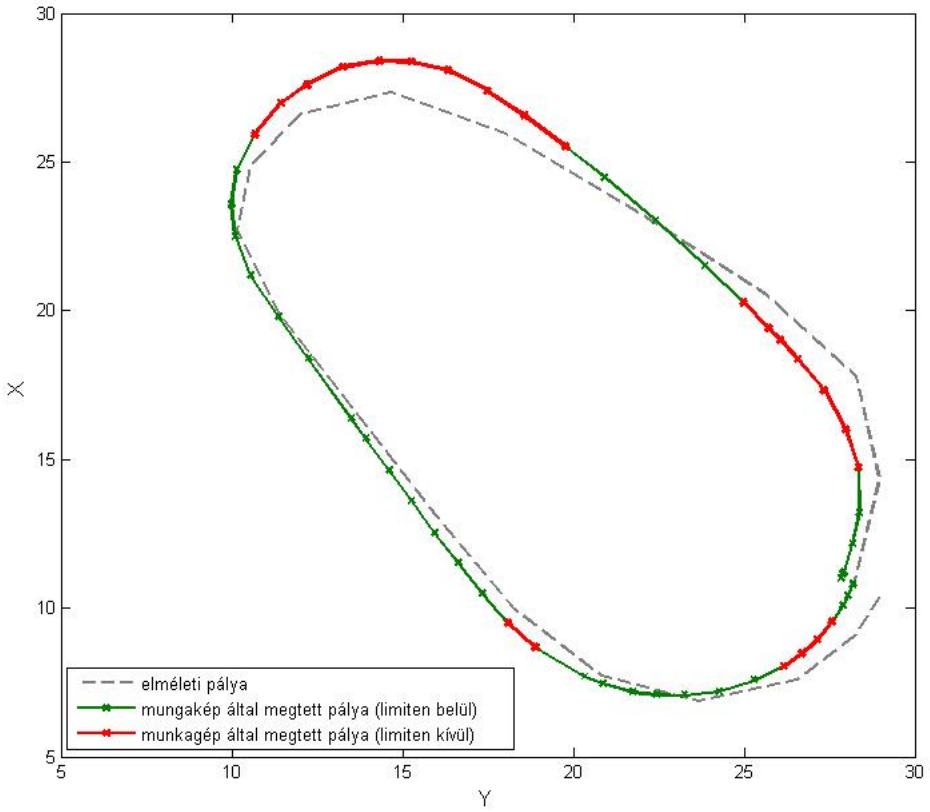


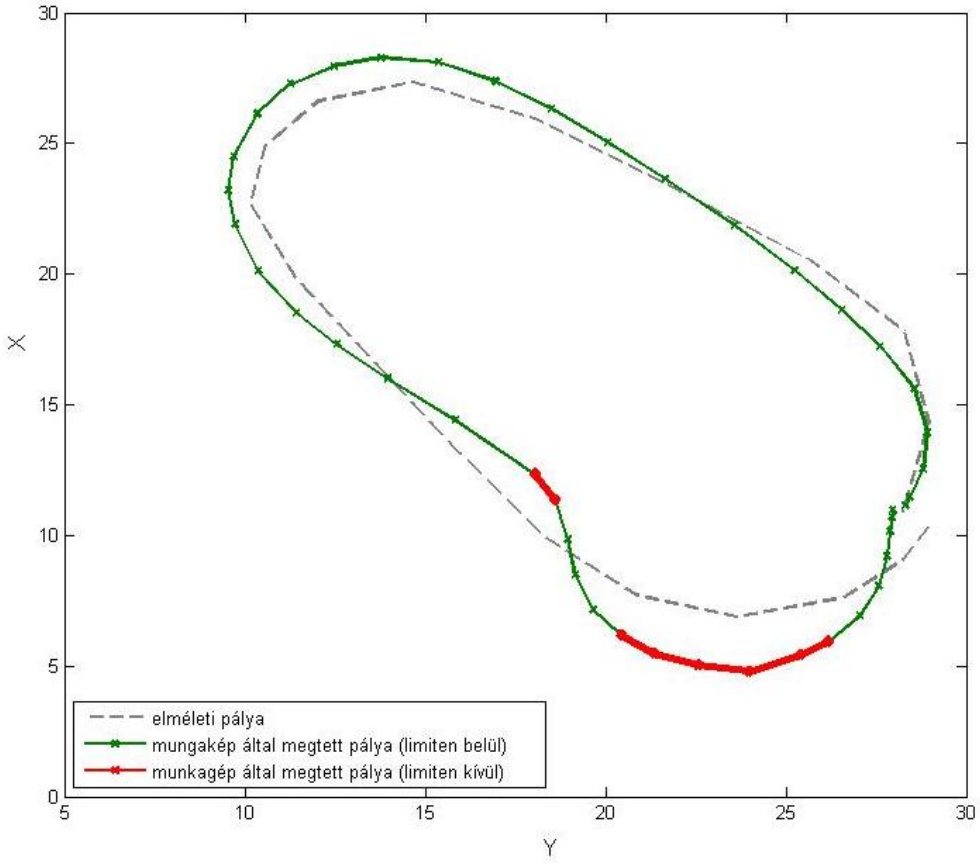
A robot mérőállomás hálózataként egy helyi koordinátarendszert vettünk fel. Elsőnek a *coofileMaker.tcl* alkalmazásával bemértük a terepen az általunk kialakított pálya nyomvonalát, mely a későbbiekben az elméleti pályaként szolgált. Ezt követően pedig mind az irányzás mind pedig a távmérés módját követés módra beállítva, valamint az első mérést elvégezve a TRACK mód aktiválása végett, elindítottuk a *measureToMachine.tcl* programot. A program futása közben a robot mérőállomás a szervomotorok segítségével szüntelenül követte a „munkagép”-re

felszerelt 360°-os prizmát. Ami a program hatékonyságát illeti, 1.4 másodpercenként tudott friss adattal szolgálni a járművel kapcsolatban. A „munkagép” vezérlésének a szimulálása során több mérés is végeztük, melyek során minden esetben a „munkagép” vezetője próbálta (vagy épp ellenkezőleg) követni a *coofileMaker.tcl* által kialakított pályánkat. Összesen 6 darab mérési sorozatot végeztünk különböző limitek és tolerancia értékek beállításával.

A program teljesítőképességét tekintve még nem teljes, mivel csak a robot mérőállomás mellett elhelyezkedő személy tudja észlelni a program eredményeképpen létre jövő információkat illetve jelzéseket. Mindenképp szükség van az adatok továbbítására a munkagép fedélzeti számítógépe felé, hogy az azt vezető képes legyen a helyes útra irányítani a járművet. Az adatok továbbítását éles esetekben általában rádión keresztül szokták megoldani.

A mérési adatok feldolgozását és megjelenítését Octave segítségével végeztem el. Az következő ábrák közül az első egy 0.5m-es toleranciával a második pedig 1m-es limittel dolgozó program eredményeinek a feldolgozását mutatja:





5.4. A rendszer továbbfejlesztése

Az Ulyxes projekt használata során számos új fejlesztéssel bővült rendszer, melyek legfőbbképp a TclAPI-t érintették. A programok tesztelését csak a Leica típusú műszereken hajtottam végre, bár egyes fejlesztések a Trimble típusú műszerekre is érvényesek. A programok használata során feltárt hibák is javításra kerültek. Az véghezvitt fejlesztések és javítások a következők voltak:

- TclAPI-hoz felhasználói dokumentáció készítése;
- A TclAPI számos új programmal kiegészült, melyek mind az automatizált mérések elvégzésével kapcsolatosak;
    - *coofileMaker.tcl*: GeoEasy-vel kompatibilis koordinátalista készítő alkalmazás;
    - *scanfileMaker.tcl*: a *robot.tcl* paraméterének a készítését automatizáló program;
    - *measureToPrism.tcl*: automatizált prizmamérést lehetővé tevő és dokumentáló program;
    - *measureToMachine.tcl*: munkagép irányítását és dokumentálását lehetővé tevő program;
- *calc.tcl* kiegészítése egyenes egyenletét valamint pont és egyenes távolságát számító függvénnyel;

- Vakon tájékozás: a program automatizált prizmakeresését és tájékozási szög számítása meg lett oldva;
- az Measure függvényhez tartozó várakozási idő (wait-time) meghosszabbítása 12 másodperce. Ennek következtében a reflektor nélküli távmérővel rendelkező műszerek robot üzemmódban 80m-nél nagyobb távolságra is képesek mérést elvégezni;
- a globális változók használatával kapcsolatos hibák javítása;
- az egyes procedúrák esetén a hibakezelés megoldásra került.

A Ulyxes rendszert számos egyéb új funkcióval lehetne még bővíteni és továbbfejleszteni. Ezek közül néhányat megemlítve:

- Az API teljes újragondolása, objektumorientált programozási nyelvek és eszközök segítségével;
- Robot mérőállomások vezérlése okos telefonom keresztül;
- Soros vonali kapcsolat helyett Bluetooth-on történő adatátvitel létrehozása;
- Az egyes programok illetve függvények tesztelése Trimble típusú műszerekkel, valamint trimble.tcl továbbfejlesztése;
- GPS mérésekre alkalmas műszerek beintegrálása az rendszerbe.

Összefoglalás:

A szakdolgozatom írásának célja arra irányult, hogy az olvasó számára betekintést nyújtsak a robot mérőállomások által kialakított automatizált mozgásvizsgálati rendszerek működésébe, valamint hogy ismertessem az BME Általános és Felsőgeodézia tanszékén kialakított és fejlesztett Ulyxes projektet. Az Ulyxes egy olyan megvalósítás, mely kizárólag nyílt forráskódú eszközök használatával valósul meg, és melynek a fejlesztése a mai napig tart a készítő, szakdolgozatokat író és végül, de nem utolsó sorban az önkéntes mérnökök által. A rendszer fő célja, hogy a robot mérőállomások automatizált vezérlésével történő méréseket, az ezekhez alkalmas adatbázisban való tárolási és kezelő műveleteken keresztül az interneten bárki számára publikálni tudjuk.

A szakdolgozatom elemi részét képező, robot mérőállomások által nyújtott automatizált méréseinket lehetővé tevő TclAPI használatával és továbbfejlesztésével számos informatikai és mérnökgeodéziai tudásra tettem szert, melyek mind a szakmai tudásomat és rálátásomat fejlesztették. A munkám során a feladatom kitért a műszerek programozásától kezdve, a mérések elvégzésén keresztül az adatok feldolgozásáig, valamint a végén kis betekintést is nyertem az adatok publikálásával kapcsolatos feladatokba is.

A rendszerrel kapcsolatban számos lehetőség nyílik azok számára, akik mind a platform mind pedig a web-programozásban jártasak, valamint erőt és inspirációt éreznek a rendszer továbbfejlesztésével kapcsolatban, mely esetén a tanszék szeretettel fogadja az érdeklődőket.

Irodalomjegyzék

[DCSEP] Dr. Csepregi Szabolcs: Mérőállomások. Segédlet, NyME GEO, 2003

[KRAU] Krauter András (2007): Geodézia. Műegyetemi Kiadó.

[AMCH] A. M. Chandra Higher Surveying - New Age International Ltd, Second edition 2005 – 424 oldal

[BBWK] Brent B. Welch, Ken Jones - Practical Programming in Tcl and Tk – Pearson Education Inc. 2003 – 882 oldal

[MAZA] Matt Zandstra : Tanuljuk meg a PHP4 használatát 24 óra alatt, Sams 2000 – 488 oldal

[TARP] Tarsoly Péter Nyugat-magyarországi egyetem – Mérőállomások 2010 http://www.tankonyvtar.hu/hu/tartalom/tamop425/0027_GED14/ch01s06.html

[RMPH] Alagútmérés, automatizált mérésfeldolgozás órai segédlet – Robot mérőállomások programozása http://www.agt.bme.hu/

[ERIC] Leica TCA1800, TCA2003, TC2003 http://eric-tang.com.hk/TCA2003.pdf

[FKKI] Fehérvári Arnold – Kallós Gábor – Kuti József Informatika 2006

[JAMA] James A. Lutes: Automated dam displacement monitoring using a robotic total station University of New Brunswick 2002 Február

[GEOMS GeoMoS Structural Engineer Journal 2002 July

[ÉPMO] Építésirányítások, mozgásvizsgálatok című tárgyhoz kapcsolódó összevont segédlet
[SOLD] www.soldata.hu
[LEIC] http://www.leica-geosystems.com/

[MTMM]http://www.mtmmagazin.hu/cikk.php?cikk_id=502&PHPSESSID=b3e0fb4e3ac6390d217b17059fbb9419

[CONS] Construction Equipment http://www.constructionequipment.com/

[AUSS] http://www.sc-vermessung.de/Ausstattung/ausstattung.html

[JFOX] Jhon Fox Applied Regression Analysis, Linear Models, and Related methods, SAGE Publications 1997 - 624 oldal

[GEPN] http://gepnet.hu/index.php?func=hirek&hea=3&art=449&ny=1

[FIGN] http://www.fig.net/pub/fig2007/papers/ts_1f/ts01f_02_tang_et_al_1326_rev.pdf

Mellékletek

scanfileMaker.tcl:

```
# !/bin/sh
# the next line restarts using tclsh \
exec tclsh "$0" "$@"

#//#
#   This program stands for making scan file from measured points.The output
file will be .tca type.
#   <p> This .tca file can be used as parameter for robot.tcl program.</p>
#
#   <p>Ulyses - an open source project to drive total stations and
#           publish observation results</p>
#   <p>GPL v2.0 license</p>
#   <p>Copyright (C) 2010-2012 Zoltan Siki <siki@agt.bme.hu></p>
#   @author Daniel Moka
#   @version 1.1
#//#


source global.tcl
source calc.tcl
source tca1800.tcl

global argv argc coo

set com "tca1800.com"
if {$argc > 0} {set com [lindex $argv 0]}

#For more detailed debugging
set debuglevel 2

# Open and set communication port
OpenCom $com

# Create and open the result file to append data
set resultFile [open resultFile.tca w+]

#Get the height of the instrument(GeoCom command)
puts  "Enter the Height of Instrument in m dimension:"
set heightOfInst [gets stdin]
#Ask the user for the station standpoint
puts  "Enter the number of the station stand point :"
set stationPointNumber [gets stdin]

# Ask the user for measuring faces (1 or 2)
puts  "Enter the faces you want to measure in your task :"
set face [gets stdin]

# Ask the user if he wants measure in RL or to prism
puts  "Are you going to measure RL(ReflectorLess) or to a prism: RL - 0 ,
prism - 1 "
set measureType [gets stdin]

#creat the first line of .tca file
puts $resultFile "!,$stationPointNumber,$heightOfInst"

while {1} {
    puts "Enter the ID of the point(only numbers allowed):"
    set pn [gets stdin]
    if {[string length $pn] == 0} {
```

```
        close $resultFile
        break
    }
    set res [Measure]
    puts $resultFile "$pn,[DMS [GetVal 7 $res]],[DMS [GetVal 8 $res]],[GetVal
9 $res],$face,$measureType"
}
```

coofileMaker.tcl

```
# !/bin/sh
# the next line restarts using tclsh \
exec tclsh "$0" "$@"

#//#
#   This program stands for creating .coo files, from measure points. Coo
file can be used as parameter at measureToMachine program, and also can be
used at
#   <p> GeoEasy for analysing.[NOTE: only start the porogram when the first
point is targeted</p>
#   <p> This .tca file can be used as parameter for robot.tcl program.</p>
#
#   <p>Ulyses - an open source project to drive total stations and
#           publish observation results</p>
#   <p>GPL v2.0 license</p>
#   <p>Copyright (C) 2010-2012 Zoltan Siki <siki@agt.bme.hu></p>
#   @author Daniel Moka
#   @version 1.1
#//#


source global.tcl
source calc.tcl
source tca1800.tcl

global argv argc coo

set com "tca1800.com"
# Adding the 1st parameter
if {$argc > 0} {set com [lindex $argv 0]}

#For more detailed debugging
set debuglevel 2

# Open and set communication port
OpenCom $com

puts "Enter the name of coo file (Note:type .coo at the end):"
set nameOfFile [gets stdin]

set outfile [open $nameOfFile w+]
set counter 1
while {1} {
    puts "Enter the ID of the point(only numbers allowed):"
    set id [gets stdin]
    if {[string length $id] == 0} {
        close $outfile
        break
    }
    set res [Measure]
    set res [Coords]
    if {[llength $res] > 0} {
        puts $outfile "{5 $id} {38 [GetVal 38 $res]} {37 [GetVal 37 $res]}
{39 1}"
    }
}
```

measureToPrism.tcl:

```
# !/bin/sh
# the next line restarts using tclsh \
exec tclsh "$0" "$@"

#//#
#   This program stands for measuring prism, recording the time and the
result of measurement
#
#   <p>Ulyses - an open source project to drive total stations and
#           publish observation results</p>
#   <p>GPL v2.0 license</p>
#   <p>Copyright (C) 2010-2012 Zoltan Siki <siki@agt.bme.hu></p>
#   @author Daniel Moka
#   @version 1.1
#//#

source global.tcl
source calc.tcl
source tca1800.tcl

global argv argc

# Adding parameter  0/1 =  measuring without distance(only angles) /
measuring angles and distances too
set dist 1
if {$argc > 0 && [lindex $argv 0] == 0} { set dist 0 }

# Adding the 2nd parameter
set com "tca1800.com"
if {$argc > 1} {set com [lindex $argv 1]}

#For more detailed debugging
set debuglevel 2

# Open and set communication port
OpenCom $com


#(Orientation of station is supposed)
while {1} {
    if {$dist} {
        set res [Measure]  ;
    } else {
        set res [GetAngles]
    }
    set systemTime [clock seconds]  ;# set current time
    puts "[GetVal 7 $res] [GetVal 8 $res] [GetVal 9 $res] [clock format
$systemTime -format %H:%M:%S]" ;# puts datas in output file
#   flush $outfile  ;#Flushes output that has been buffered for outputfile
channelId.

}

close $outfile ;
```

measureToMachine.tcl:

```
# !/bin/sh
# the next line restarts using tclsh \
exec tclsh "$0" "$@"

#//#
#   This program stands for measuring a machine, while scanning the path of
machine and warns if the machine is on wrong way.
#
#   <p>Ulyses - an open source project to drive total stations and
#           publish observation results</p>
#   <p>GPL v2.0 license</p>
#   <p>Copyright (C) 2010-2012 Zoltan Siki <siki@agt.bme.hu></p>
#   @author Daniel Moka
#   @version 1.1
#//#

source global.tcl
source calc.tcl
source tca1800.tcl

global argv argc coo

set com "tca1800.com"

# Adding the 1st parameter
if {$argc > 0} {set limit [lindex $argv 0]}

# Adding the 2nd parameter
if {$argc > 1} {set tol [lindex $argv 1]}

# Adding the 3nd parameter
if {$argc > 2} {set cooFile [lindex $argv 2]}


#For more detailed debugging
set debuglevel 0

# Open and set communication port
OpenCom $com

#f {[LoadCoo "../../../Munkagep/SavoyaNov28.coo"] != 0} {
    #puts "Error in input file"
    #return
#}
LoadCoo "../../../Munkagep/SavoyaNov28.coo"

set outfile [open "measuretomachine3.log" w+]


while {1} {
set t0 [clock clicks]
    set systemTime [clock seconds]  ;# set current time
    set res [Coords]
        if {[llength $res] > 1} {
            set y [GetVal 38 $res]
            set x [GetVal 37 $res]
            puts $outfile "x: $x y: $y"
            set mind 1e38
            set yj [GetVal 38 $coo(1)]
```

```
            set xj [GetVal 37 $coo(1)]
            for {set j 2} {$j <= [array size coo]} {incr j} {
                set yi $yj
                set xi $xj
                set yj [GetVal 38 $coo($j)]
                set xj [GetVal 37 $coo($j)]
                set miny [expr {$yi < $yj ? $yi - $tol : $yj - $tol}]
                set minx [expr {$xi < $xj ? $xi - $tol : $xj - $tol}]
                set maxy [expr {$yi > $yj ? $yi + $tol : $yj + $tol}]
                set maxx [expr {$xi > $xj ? $xi + $tol : $xj + $tol}]
                if {$minx < $x && $x < $maxx && $miny < $y && $y < $maxy } {
                    set l [Line2D $xi $yi $xj $yj] ;# calculates the equation
of a line
                    set d [LinePointDist [lindex $l 0] [lindex $l 1] [lindex
$l 2] $x $y ] ;# calculates distance between a point and a line
                    puts $outfile "dist: $d  i: [expr {$j-1}]  j:$j"
                    if {$d < $mind} {
                        set mind $d
                    }
                }
            }
            puts $outfile "minD: $mind"
            if {$mind > $limit} {
                if {$mind==1e38} {
                    set mind "N/A"
                }
                puts $outfile "The machine is not on the right path on y: $y
x: $x coordinates,distance : $mind at [clock format $systemTime -format
%H:%M:%S]"
                flush $outfile  ;#Flushes output that has been buffered for
outputfile channelId.
                for {set k 1} {$k < 5} {incr k} {
                    puts [format "%c" 7] ;#short warning
                }
            } else {
                puts $outfile "The machine is on the right path on y: $y x:
$x coordinates,distance : $mind at [clock format $systemTime -format
%H:%M:%S]"
                flush $outfile  ;#Flushes output that has been buffered for
outputfile channelId.
            }
        } else {
                puts $outfile "Error : $res at [clock format $systemTime -
format %H:%M:%S]"
                flush $outfile  ;#Flushes output that has been buffered for
outputfile channelId.
        }
puts "ido: [expr {[clock clicks] - $t0}]"
}


close $outfile
```

merge.tcl:

```
set fbal [open "balOldalResults/hidMeresBalOldal108.txt" r]
set fjobb [open "jobbOldalResults/hidMeresJobbOldal108.txt" r]
set fm [open "merged.txt" w]
set bufbal [gets $fbal]
set listbal [split $bufbal]
set bufjobb [gets $fjobb]
set listjobb [split $bufjobb]
while {! [eof $fbal] && ! [eof $fjobb]} {
    if {[lindex $listbal 3] == [lindex $listjobb 3]} {
        puts $fm "[lindex $listbal 3];[lindex $listbal 0];[lindex $listbal
1];[lindex $listbal 2];[lindex $listjobb 0];[lindex $listjobb 1];[lindex
$listjobb 2]"
        set bufbal [gets $fbal]
        set listbal [split $bufbal]
        set bufjobb [gets $fjobb]
        set listjobb [split $bufjobb]
    } elseif {[lindex $listbal 3] < [lindex $listjobb 3]} {
        puts $fm "[lindex $listbal 3];[lindex $listbal 0];[lindex $listbal
1];[lindex $listbal 2];;;"
        set bufbal [gets $fbal]
        set listbal [split $bufbal]
    } else {
        puts $fm "[lindex $listjobb 3];;;;[lindex $listjobb 0];[lindex
$listjobb 1];[lindex $listjobb 2]"
        set bufjobb [gets $fjobb]
        set listjobb [split $bufjobb]
    }
}

close $fbal
close $fjobb
close $fm
```

Z épület bemért pontjaira illesztett síkok:

Szeptember 26 T=24° (38-Y 37-X 39-Z) 2012.11.24 15:00 - Sík y,x és z változik

z = -339619.674 + 31.66410849 * y + 3454.53889772 * x
Lejtõ irány: 0-31-31 Lejtõ szög: 89-59-00

Pontszám	y	x	z	távolság	dy	dx	dz
100	-56.277	98.833	37.112	0.005	0.000	0.005	-0.000
101	-48.783	98.764	37.116	0.005	0.000	0.005	-0.000
102	-41.286	98.693	37.110	0.008	0.000	0.008	-0.000
103	-33.770	98.627	37.113	0.005	0.000	0.005	-0.000
104	-26.689	98.569	37.130	-0.003	-0.000	-0.003	0.000
105	-56.230	98.843	33.668	-0.007	-0.000	-0.007	0.000
106	-48.783	98.772	33.670	-0.004	-0.000	-0.004	0.000
107	-41.288	98.709	33.667	-0.010	-0.000	-0.010	0.000
108	-33.782	98.629	33.674	0.002	0.000	0.002	-0.000
109	-26.698	98.566	33.670	-0.000	-0.000	-0.000	0.000
110	-56.293	98.831	30.243	0.004	0.000	0.004	-0.000
111	-48.793	98.766	30.227	0.001	0.000	0.001	-0.000
112	-41.290	98.700	30.225	-0.002	-0.000	-0.002	0.000
113	-33.782	98.631	30.237	-0.001	-0.000	-0.001	0.000
114	-26.705	98.557	30.235	0.007	0.000	0.007	-0.000
115	-56.281	98.830	26.797	0.005	0.000	0.005	-0.000
116	-48.784	98.770	26.802	-0.004	-0.000	-0.004	0.000
117	-41.282	98.703	26.800	-0.005	-0.000	-0.005	0.000
118	-33.777	98.628	26.793	0.000	0.000	0.000	-0.000
119	-26.699	98.566	26.789	-0.002	-0.000	-0.002	0.000
120	-48.776	98.762	23.357	0.003	0.000	0.003	-0.000
121	-41.278	98.697	23.366	-0.001	-0.000	-0.001	0.000
122	-33.786	98.628	23.356	-0.000	-0.000	-0.000	0.000
123	-26.711	98.558	23.374	0.005	0.000	0.005	-0.000
124	-48.768	98.763	19.921	0.001	0.000	0.001	-0.000
125	-41.270	98.698	19.924	-0.003	-0.000	-0.003	0.000
126	-33.782	98.634	19.911	-0.007	-0.000	-0.007	0.000
127	-26.710	98.565	19.926	-0.004	-0.000	-0.004	0.000
128	-41.278	98.695	16.477	-0.001	-0.000	-0.001	0.000
129	-33.786	98.626	16.491	-0.001	-0.000	-0.001	0.000
130	-26.716	98.560	16.489	0.001	0.000	0.001	-0.000
131	-41.276	98.702	13.049	-0.009	-0.000	-0.009	0.000
132	-33.780	98.632	13.037	-0.008	-0.000	-0.008	0.000
133	-41.287	98.688	9.615	0.004	0.000	0.004	-0.000
134	-33.779	98.623	9.616	0.000	0.000	0.000	-0.000
135	-40.704	98.689	5.802	-0.003	-0.000	-0.003	0.000
136	-33.593	98.602	5.809	0.018	0.000	0.018	-0.000

RMS=0.005

Oktober 4. T=14° 2012.11.24 15:02 - Sík y,x és z változik

z = -327147.236 + 30.42763148 * y + 3327.70273230 * x

Lejtõ irány: 0-31-26 Lejtõ szög: 89-58-58

Pontszám	y	x	z	távolság	dy	dx	dz
100	-56.279	98.832	37.111	0.004	0.000	0.004	-0.000
101	-48.782	98.763	37.113	0.005	0.000	0.005	-0.000
102	-41.284	98.690	37.109	0.009	0.000	0.009	-0.000
103	-33.770	98.625	37.108	0.005	0.000	0.005	-0.000
104	-26.689	98.569	37.119	-0.004	-0.000	-0.004	0.000
105	-56.227	98.841	33.669	-0.007	-0.000	-0.007	0.000
106	-48.784	98.773	33.666	-0.006	-0.000	-0.006	0.000
107	-41.285	98.707	33.667	-0.010	-0.000	-0.010	0.000
108	-33.780	98.627	33.672	0.002	0.000	0.002	-0.000
109	-26.701	98.566	33.669	-0.001	-0.000	-0.001	0.000
110	-56.293	98.830	30.240	0.004	0.000	0.004	-0.000
111	-48.792	98.764	30.229	0.001	0.000	0.001	-0.000
112	-41.292	98.699	30.232	-0.002	-0.000	-0.002	0.000
113	-33.782	98.629	30.233	-0.000	-0.000	-0.000	0.000
114	-26.704	98.555	30.231	0.008	0.000	0.008	-0.000
115	-56.280	98.828	26.794	0.005	0.000	0.005	-0.000
116	-48.783	98.767	26.799	-0.003	-0.000	-0.003	0.000
117	-41.281	98.700	26.797	-0.004	-0.000	-0.004	0.000
118	-33.776	98.625	26.792	0.002	0.000	0.002	-0.000
119	-26.700	98.565	26.786	-0.003	-0.000	-0.003	0.000
120	-48.777	98.762	23.354	0.002	0.000	0.002	-0.000
121	-41.280	98.698	23.361	-0.003	-0.000	-0.003	0.000
122	-33.789	98.626	23.358	0.000	0.000	0.000	-0.000
123	-26.711	98.556	23.371	0.005	0.000	0.005	-0.000
124	-48.766	98.760	19.917	0.002	0.000	0.002	-0.000
125	-41.270	98.696	19.918	-0.002	-0.000	-0.002	0.000
126	-33.781	98.630	19.918	-0.005	-0.000	-0.005	0.000
127	-26.711	98.564	19.923	-0.004	-0.000	-0.004	0.000
128	-41.279	98.694	16.479	-0.002	-0.000	-0.002	0.000
129	-33.784	98.624	16.482	0.000	0.000	0.000	-0.000
130	-26.716	98.561	16.490	-0.002	-0.000	-0.002	0.000
131	-41.275	98.700	13.046	-0.008	-0.000	-0.008	0.000
132	-33.780	98.631	13.036	-0.008	-0.000	-0.008	0.000
133	-41.284	98.686	9.611	0.005	0.000	0.005	-0.000
134	-33.780	98.622	9.611	0.000	0.000	0.000	-0.000
135	-40.704	98.687	5.802	-0.003	-0.000	-0.003	0.000
136	-33.593	98.602	5.807	0.017	0.000	0.017	-0.000

RMS=0.005

Október 30° T=4° 2012.11.24 15:04 - Sík y,x és z változik

z = -384665.001 + 35.66794069 * y + 3912.69035213 * x

Lejtõ irány: 0-31-20 Lejtõ szög: 89-59-07

Pontszám	y	x	z	távolság	dy	dx	dz	
100	-56.273	98.828	37.110	0.006	0.000	0.006	-0.000	
101	-48.784	98.760	37.115	0.006	0.000	0.006	-0.000	
102	-41.285	98.692	37.113	0.006	0.000	0.006	-0.000	
103	-33.771	98.624	37.110	0.005	0.000	0.005	-0.000	
104	-26.690	98.567	37.121	-0.002	-0.000	-0.002	0.000	
105	-56.228	98.841	33.670	-0.007	-0.000	-0.007	0.000	
106	-48.784	98.771	33.666	-0.005	-0.000	-0.005	0.000	
107	-41.286	98.706	33.667	-0.009	-0.000	-0.009	0.000	
108	-33.781	98.629	33.673	-0.000	-0.000	-0.000	0.000	
109	-26.698	98.566	33.671	-0.002	-0.000	-0.002	0.000	
110	-56.293	98.830	30.242	0.003	0.000	0.003	-0.000	
111	-48.791	98.764	30.231	0.001	0.000	0.001	-0.000	
112	-41.288	98.699	30.231	-0.003	-0.000	-0.003	0.000	
113	-33.782	98.628	30.236	0.000	0.000	0.000	-0.000	
114	-26.706	98.556	30.234	0.008	0.000	0.008	-0.000	
115	-56.279	98.829	26.798	0.003	0.000	0.003	-0.000	
116	-48.782	98.768	26.799	-0.004	-0.000	-0.004	0.000	
117	-41.281	98.697	26.802	-0.002	-0.000	-0.002	0.000	
118	-33.777	98.625	26.794	0.001	0.000	0.001	-0.000	
119	-26.699	98.565	26.786	-0.003	-0.000	-0.003	0.000	
120	-48.776	98.760	23.355	0.003	0.000	0.003	-0.000	
121	-41.278	98.697	23.361	-0.002	-0.000	-0.002	0.000	
122	-33.789	98.626	23.360	0.000	0.000	0.000	-0.000	
123	-26.712	98.556	23.372	0.005	0.000	0.005	-0.000	
124	-48.767	98.760	19.918	0.002	0.000	0.002	-0.000	
125	-41.271	98.695	19.922	-0.002	-0.000	-0.002	0.000	
126	-33.784	98.630	19.918	-0.005	-0.000	-0.005	0.000	
127	-26.710	98.564	19.925	-0.003	-0.000	-0.003	0.000	
128	-41.279	98.693	16.480	-0.001	-0.000	-0.001	0.000	
129	-33.787	98.625	16.484	-0.000	-0.000	-0.000	0.000	
130	-26.715	98.560	16.488	-0.000	-0.000	-0.000	0.000	
131	-41.275	98.700	13.048	-0.008	-0.000	-0.008	0.000	
132	-33.782	98.632	13.037	-0.009	-0.000	-0.009	0.000	
133	-41.285	98.686	9.614	0.005	0.000	0.005	-0.000	
134	-33.781	98.623	9.613	-0.001	-0.000	-0.001	0.000	
135	-40.704	98.687	5.801	-0.002	-0.000	-0.002	0.000	
136	-33.595	98.604	5.808	0.016	0.000	0.016	-0.000	

RMS=0.005

Gömb illesztése parabolára
2012.11.06 11:02 - Gömb
Y0 = 2.388 X0 = 0.503 Z0 = 4.005 R = 0.309

Pontszám	y	x	z	dy	dx	dz	dr
3	2.220	0.636	4.235	0.003	-0.002	-0.004	-0.005
4	2.215	0.644	4.226	0.002	-0.002	-0.004	-0.005
5	2.230	0.601	4.255	0.001	-0.001	-0.002	-0.002
6	2.238	0.556	4.270	-0.000	0.000	0.000	0.000
7	2.238	0.565	4.270	0.000	-0.000	-0.001	-0.001
8	2.242	0.518	4.277	-0.000	0.000	0.000	0.000
9	2.241	0.471	4.275	-0.000	-0.000	0.000	0.000
10	2.241	0.481	4.275	-0.001	-0.000	0.001	0.001
11	2.234	0.432	4.263	-0.001	-0.000	0.001	0.001
12	2.219	0.383	4.234	-0.000	-0.000	0.000	0.000
13	2.222	0.393	4.241	-0.001	-0.000	0.001	0.001
14	2.207	0.343	4.211	0.005	0.004	-0.006	-0.008
21	2.201	0.633	4.219	0.002	-0.001	-0.002	-0.003
22	2.197	0.641	4.212	0.002	-0.002	-0.003	-0.004
23	2.212	0.598	4.240	-0.001	0.000	0.001	0.001
24	2.231	0.556	4.277	0.004	-0.001	-0.008	-0.009
25	2.229	0.565	4.272	0.003	-0.001	-0.006	-0.007
26	2.233	0.518	4.280	0.003	-0.000	-0.006	-0.007
27	2.226	0.470	4.266	-0.000	-0.000	0.000	0.000
28	2.228	0.480	4.270	0.000	0.000	-0.001	-0.001
29	2.218	0.430	4.251	-0.001	-0.000	0.001	0.001
30	2.202	0.381	4.221	0.000	0.000	-0.001	-0.000
31	2.206	0.391	4.228	-0.000	-0.000	0.000	0.000
38	2.257	0.684	4.225	0.001	-0.002	-0.003	-0.004
39	2.273	0.642	4.256	-0.000	0.001	0.001	0.001
40	2.270	0.650	4.249	-0.001	0.001	0.001	0.001
41	2.285	0.606	4.277	-0.000	0.000	0.001	0.001
42	2.293	0.561	4.292	-0.001	0.000	0.002	0.002
43	2.292	0.570	4.290	-0.001	0.000	0.001	0.001
44	2.296	0.523	4.298	-0.001	0.000	0.001	0.001
45	2.296	0.476	4.298	-0.001	-0.000	0.001	0.001
46	2.296	0.485	4.298	-0.001	-0.000	0.002	0.002
47	2.292	0.436	4.290	-0.000	-0.000	0.001	0.001
48	2.281	0.388	4.270	-0.001	-0.001	0.001	0.001
49	2.284	0.397	4.275	-0.001	-0.001	0.001	0.001
50	2.266	0.347	4.242	-0.000	-0.000	0.000	0.001
55	2.297	0.732	4.203	0.002	-0.004	-0.004	-0.006
56	2.318	0.690	4.240	-0.000	0.001	0.000	0.001
57	2.335	0.648	4.272	-0.000	0.001	0.001	0.001
58	2.332	0.656	4.267	-0.000	0.001	0.001	0.001
59	2.345	0.611	4.290	-0.000	0.001	0.001	0.002
60	2.352	0.566	4.304	-0.000	0.000	0.002	0.002
61	2.351	0.575	4.302	-0.000	0.001	0.002	0.002
65	2.352	0.440	4.304	-0.000	-0.000	0.002	0.002
66	2.344	0.391	4.289	-0.000	-0.001	0.001	0.001

67	2.346	0.401	4.293	-0.000	-0.001	0.001	0.001
68	2.332	0.351	4.267	-0.000	-0.001	0.001	0.001
69	2.310	0.301	4.227	0.000	0.000	-0.001	-0.000
70	2.315	0.311	4.236	-0.000	-0.001	0.000	0.000
74	2.306	0.689	4.238	-0.000	0.001	0.000	0.001
75	2.323	0.647	4.270	-0.000	0.001	0.001	0.001
76	2.320	0.655	4.264	-0.000	0.001	0.001	0.002
77	2.333	0.610	4.288	-0.000	0.001	0.002	0.002
78	2.341	0.565	4.302	-0.001	0.001	0.003	0.003
79	2.340	0.574	4.301	-0.000	0.000	0.001	0.001
80	2.344	0.526	4.307	-0.001	0.000	0.003	0.003
83	2.341	0.439	4.302	-0.000	-0.000	0.002	0.002
84	2.332	0.391	4.286	-0.000	-0.001	0.001	0.001
85	2.335	0.400	4.291	-0.000	-0.000	0.001	0.001
86	2.320	0.350	4.264	-0.000	-0.001	0.001	0.001
87	2.297	0.301	4.221	0.000	0.000	-0.001	-0.001
88	2.302	0.310	4.231	-0.000	-0.000	-0.000	0.000
91	2.343	0.736	4.208	0.000	-0.002	-0.002	-0.003
92	2.365	0.694	4.247	-0.000	0.001	0.000	0.001
93	2.381	0.651	4.275	-0.000	0.001	0.001	0.002
94	2.377	0.659	4.270	-0.000	0.001	0.002	0.002
96	2.397	0.568	4.305	0.000	0.001	0.002	0.002
97	2.396	0.577	4.303	0.000	0.001	0.002	0.002
98	2.400	0.529	4.310	0.000	0.000	0.003	0.003
99	2.400	0.482	4.311	0.000	-0.000	0.003	0.003
101	2.398	0.442	4.307	0.000	-0.000	0.001	0.001
102	2.390	0.394	4.292	0.000	-0.001	0.002	0.002
103	2.393	0.403	4.297	0.000	-0.000	0.001	0.001
104	2.380	0.353	4.274	-0.000	-0.001	0.001	0.001
105	2.361	0.303	4.239	-0.000	-0.001	0.000	0.000
106	2.365	0.313	4.248	-0.000	-0.001	0.000	0.000
109	2.396	0.740	4.208	-0.000	-0.002	-0.002	-0.003
110	2.416	0.697	4.244	0.000	0.001	0.000	0.001
111	2.432	0.654	4.270	0.000	0.001	0.001	0.002
112	2.429	0.662	4.265	0.000	0.001	0.002	0.002
113	2.441	0.617	4.288	0.000	0.000	0.000	0.000
114	2.448	0.570	4.299	0.000	0.001	0.002	0.002
115	2.447	0.580	4.297	0.000	0.001	0.002	0.002
116	2.452	0.532	4.306	0.000	0.000	0.001	0.001
117	2.452	0.484	4.306	0.000	-0.000	0.001	0.002
118	2.452	0.493	4.305	0.001	-0.000	0.003	0.003
119	2.449	0.444	4.301	0.000	-0.000	0.001	0.002
120	2.442	0.395	4.289	0.000	-0.001	0.001	0.001
121	2.444	0.405	4.292	0.000	-0.001	0.001	0.001
122	2.432	0.355	4.272	0.000	-0.001	0.001	0.001
123	2.415	0.305	4.241	0.000	-0.001	0.000	0.001
124	2.419	0.315	4.248	0.000	-0.001	0.000	0.001
125	2.396	0.264	4.207	-0.000	0.002	-0.003	-0.004
127	2.386	0.739	4.209	0.000	-0.002	-0.002	-0.002
128	2.406	0.697	4.244	0.000	0.001	0.001	0.001

129	2.422	0.653	4.272	0.000	0.001	0.001	0.002
130	2.419	0.662	4.267	0.000	0.001	0.001	0.002
131	2.431	0.616	4.288	0.000	0.001	0.001	0.001
132	2.439	0.570	4.302	0.000	0.000	0.001	0.001
133	2.437	0.579	4.299	0.000	0.001	0.002	0.002
134	2.442	0.532	4.308	0.000	0.000	0.001	0.001
135	2.442	0.484	4.308	0.000	-0.000	0.001	0.001
136	2.442	0.493	4.308	0.000	-0.000	0.002	0.002
137	2.440	0.444	4.303	0.000	-0.000	0.001	0.001
138	2.433	0.395	4.291	0.000	-0.000	0.000	0.001
139	2.434	0.405	4.294	0.000	-0.001	0.001	0.001
140	2.422	0.354	4.272	0.000	-0.001	0.001	0.002
141	2.405	0.305	4.241	0.000	-0.001	0.000	0.001
142	2.409	0.315	4.249	0.000	-0.001	0.000	0.001
143	2.385	0.264	4.206	0.000	0.002	-0.002	-0.003
146	2.456	0.699	4.235	0.000	0.001	0.000	0.001
147	2.470	0.655	4.260	0.001	0.001	0.001	0.002
148	2.468	0.664	4.256	0.000	0.001	0.000	0.001
149	2.480	0.618	4.276	0.001	0.001	0.001	0.001
150	2.487	0.572	4.289	0.000	0.000	0.001	0.001
151	2.486	0.581	4.287	0.000	0.000	0.001	0.001
152	2.490	0.533	4.294	0.001	0.000	0.001	0.001
153	2.491	0.485	4.295	0.001	-0.000	0.002	0.002
154	2.492	0.495	4.297	0.000	-0.000	-0.000	-0.000
155	2.489	0.445	4.291	0.000	-0.000	0.001	0.001
156	2.481	0.396	4.279	0.000	-0.000	0.001	0.001
157	2.484	0.406	4.283	0.000	-0.000	0.000	0.000
158	2.472	0.356	4.263	0.000	-0.000	0.000	0.001
159	2.455	0.306	4.234	0.000	-0.000	-0.000	-0.000
160	2.459	0.316	4.241	0.000	-0.000	-0.000	-0.000
164	2.500	0.700	4.218	-0.000	-0.001	-0.001	-0.001
165	2.515	0.656	4.242	0.000	0.000	0.000	0.000
166	2.512	0.665	4.238	0.000	0.000	-0.000	-0.000
167	2.523	0.619	4.257	0.001	0.001	0.001	0.001
168	2.530	0.572	4.268	0.001	0.001	0.002	0.003
169	2.530	0.582	4.268	0.001	0.000	0.000	0.001
170	2.534	0.533	4.275	0.001	0.000	0.001	0.002
171	2.535	0.486	4.276	0.001	-0.000	0.001	0.001
172	2.535	0.495	4.277	0.000	-0.000	0.000	0.001
173	2.532	0.446	4.271	0.001	-0.000	0.001	0.002
174	2.525	0.397	4.260	0.001	-0.000	0.000	0.001
175	2.527	0.406	4.263	0.001	-0.001	0.001	0.001
176	2.516	0.356	4.245	0.000	-0.000	0.000	0.001
177	2.501	0.307	4.219	-0.001	0.001	-0.002	-0.002
178	2.505	0.316	4.225	-0.001	0.001	-0.002	-0.002
182	2.492	0.700	4.221	-0.000	-0.000	-0.001	-0.001
183	2.506	0.656	4.246	0.000	0.000	0.000	0.000
184	2.504	0.665	4.242	0.000	0.000	0.000	0.000
185	2.516	0.619	4.262	0.000	0.000	-0.000	0.000
186	2.522	0.572	4.274	0.001	0.000	0.001	0.001

187	2.521	0.581	4.271	0.001	0.001	0.002	0.002
188	2.526	0.533	4.279	0.001	0.000	0.001	0.001
189	2.527	0.486	4.281	0.000	-0.000	0.000	0.000
190	2.526	0.495	4.280	0.001	-0.000	0.001	0.001
191	2.524	0.446	4.277	0.000	-0.000	0.000	0.000
192	2.517	0.397	4.264	0.001	-0.000	0.001	0.001
193	2.519	0.406	4.268	0.000	-0.000	-0.000	0.000
194	2.508	0.356	4.250	0.000	-0.000	-0.000	-0.000
195	2.493	0.307	4.223	-0.001	0.001	-0.002	-0.002
196	2.496	0.316	4.229	-0.000	0.001	-0.001	-0.001
202	2.547	0.665	4.220	-0.001	-0.001	-0.002	-0.003
203	2.558	0.619	4.238	-0.000	-0.000	-0.001	-0.001
204	2.564	0.572	4.249	0.000	0.000	0.000	0.000
205	2.564	0.582	4.248	0.000	0.000	-0.000	-0.000
206	2.568	0.533	4.255	0.000	0.000	-0.000	0.000
207	2.568	0.486	4.256	0.000	-0.000	0.000	0.000
208	2.568	0.495	4.255	0.001	-0.000	0.001	0.001
209	2.566	0.446	4.253	-0.000	0.000	-0.001	-0.001
210	2.560	0.397	4.241	-0.001	0.000	-0.001	-0.001
211	2.561	0.406	4.244	-0.000	0.000	-0.001	-0.001
212	2.551	0.356	4.228	-0.001	0.001	-0.003	-0.003
222	2.603	0.572	4.222	-0.002	-0.001	-0.003	-0.003
223	2.603	0.581	4.221	-0.003	-0.001	-0.004	-0.005
224	2.606	0.533	4.227	-0.002	-0.000	-0.002	-0.003
225	2.607	0.485	4.228	-0.002	0.000	-0.002	-0.003
226	2.607	0.494	4.228	-0.002	0.000	-0.002	-0.003
227	2.605	0.445	4.225	-0.003	0.001	-0.003	-0.004
239	2.590	0.619	4.219	-0.004	-0.002	-0.005	-0.007
240	2.595	0.572	4.226	-0.000	-0.000	-0.001	-0.001
241	2.594	0.581	4.225	-0.001	-0.000	-0.001	-0.002
242	2.599	0.533	4.233	-0.001	-0.000	-0.002	-0.002
243	2.599	0.485	4.233	-0.001	0.000	-0.002	-0.002
244	2.599	0.495	4.233	-0.001	0.000	-0.001	-0.001
245	2.597	0.445	4.229	-0.001	0.000	-0.002	-0.002
246	2.593	0.397	4.222	-0.004	0.002	-0.005	-0.007
247	2.593	0.406	4.223	-0.003	0.002	-0.004	-0.005

RMS=0.002

Tcl dokumentáció – Procedure Detail: *calc.tcl*

Procedure Detail

::Bearing

```
proc  ::Bearing { ea na eb nb }
```

Bearing function: Calculate whole circle bearing counter clockwise from north

Parameters:

`xa, ya` - coordinates of station
`xb, yb` - coordinates of reference point

Returns:

bearing in radian

Defined in:

calc.tcl, line 319

::ChangeAngle

```
proc  ::ChangeAngle { angle {in DMS} {out RAD} }
```

Conversion function: Universal angle conversion function

Parameters:

`angle` - the angle to convert
`in` - actual unit of angle (DMS/DEG/RAD/GON)
`out` - target unit for result (DMS/DEG/RAD/GON)

Returns:

angle in out unit

Defined in:

calc.tcl, line 128

::Deg2Rad

```
proc  ::Deg2Rad { deg }
```

Conversion function: Convert sexagesimal angle to radian

Parameters:

`angle` - in pseudo dms format (ddd.mmss)

Returns:

angle in radians

Defined in:

calc.tcl, line 17

::DelVal

```
proc  ::DelVal { codes buf }
```

List handling function: Delete sublist from list

Parameters:

`codes` - list of codes to remove from buf
`buf` - list of pair of elements like {{code1 value1} {code2 value2} ...}

Returns:

the list without codes

Defined in:

calc.tcl, line 166

::DisplayAngles

```
proc ::DisplayAngles { anglist {unit DMS} }
```

Display angles on standard output

Parameters:

`anglist` - code list with angle values in radian {{7 hz} {8 v} ...}
`unit` - for output

Defined in:

calc.tcl, line 201

::DMS

```
proc ::DMS { val }
```

Conversion function: Convert radian to DMS (sexagesimal)

Parameters:

`val` - angle in radian

Returns:

angle in ddd-mm-ss format

Defined in:

calc.tcl, line 109

::DMS2Rad

```
proc ::DMS2Rad { dms }
```

Conversion function: Convert angle from DMS (sexagesimal) to radian

Parameters:

`angle` - in DMS (deg-min-sec) to convert into radian

Returns:

angle in radian or empty string if invalid value got

Defined in:

calc.tcl, line 75

::GetVal

```
proc ::GetVal { codes buf }
```

List handling function: Get value from list of lists like {{code1 value1} {code2 value2} ...}

Parameters:

`codes` - list of codes to look for in buf
`buf` - list of pair of elements like {{code1 value1} {code2 value2} ...}

Returns:

value which belongs to the first code from codes found in buf or empty string if none of the codes found.

Defined in:

calc.tcl, line 151

::Gon2Rad

```
proc ::Gon2Rad { angle }
```

Conversion function: Convert angle from gon to radian

Parameters:

`angle` - value in gon

Returns:

angle in radian

Defined in:

calc.tcl, line 45

::Line2D

```
proc  ::Line2D { e1 n1 e2 n2 }
```

Line2D calculates the equation of a line going through two points a * e + b * n + c = 0

Parameters:

`e1,n1` - easting and northing coordinates of startpoint
`e2,n2` - easting and northing coordiates of endpoint

Returns:

list of line coefficients {a b c}

Defined in:

calc.tcl, line 334

::LinePointDist

```
proc  ::LinePointDist { a b c e n }
```

LinePointDist calculates distance between a point and a line

Parameters:

`a,b,c` - coefficients of the equation of the line
`e,n` - easting and northing coordinates of point

Returns:

distance

Defined in:

calc.tcl, line 345

::LoadCoo

```
proc  ::LoadCoo { fn }
```

Load GeoEasy coordinate file into global array coo
WARNING previous content of coo array lost!

Returned error codes:

- -1: cannot open file
- positive value: line number with error

Parameters:

`fn` - input file name

Returns:

0 on success or nonzero error code

Defined in:

calc.tcl, line 214

::LoadGeo

```
proc  ::LoadGeo { fn }
```

Load GeoEasy fieldbook (.geo file) into memory array geo
WARNING previous content of geo array lost!

Returned error codes:

- -1: cannot open file
- positive value: line number with error

Parameters:

`fn` - file name of GeoEasy data set

Returns:

0 on success or nonzero error code

Defined in:

calc.tcl, line 262

::MoveRel

```
proc  ::MoveRel { hz_rel v_rel {units RAD} {atr 0} }
```

Instrument handling (instrument type independent functions): Rotate instrument relative to the actual position

Parameters:

`hz_rel` - relative horizontal movement, + to right, - to left
`v_rel` - relative vertical movement, + to down, - to up
`units` - input angle unit (RAD/DMS/DEG/GON), optional
`atr` - 0/1 move without ATR/move with ATR, optional

Defined in:

calc.tcl, line 182

::Rad2Deg

```
proc  ::Rad2Deg { angle }
```

Conversion function: Convert radian to sexagesimal into pseudo dms (ddd.mmss) format

Parameters:

`angle` - value in radian

Returns:

angle in pseudo DMS

Defined in:

calc.tcl, line 30

::Rad2Gon

```
proc  ::Rad2Gon { angle }
```

Conversion function: Convert angle from radian to gon

Parameters:

`angle` - angle value in radian

Returns:

angle in gon

Defined in:

calc.tcl, line 55

::Rad2Sec

```
proc  ::Rad2Sec { rad }
```

Conversion function: Convert angle from radian to seconds (ss)

Parameters:

`angle` - angle value in radian

Returns:

angle in second

Defined in:

calc.tcl, line 65

Procedure Detail

::CloseCom

```
proc  ::CloseCom {  }
```

Close communicaton port

Defined in:
tca1800.tcl, line 49

::Coords

```
proc  ::Coords { {wait 1000} {incl 0} }
```

Read coordinates from instrument calculated from last distance measurement

Parameters:
`wait` - wait time in ms, optional (default 1000)
`incl` - inclination calculation - 0/1/2 = measure always (slow)/calculate (fast)/automatic, optional (default 0)

Returns:
coordinate list : {{38 Easting} {37 Northing} {39 Height}}

Defined in:
tca1800.tcl, line 436

::GetAtmCorr

```
proc  ::GetAtmCorr {  }
```

Get atmospheric correction settings

Returns:
atmospheric settings as a list {lambda pressure drytemp wettemp}

Defined in:
tca1800.tcl, line 217

::GetATR

```
proc  ::GetATR {  }
```

Get ATR status

Returns:
0/1 - off/on

Defined in:
tca1800.tcl, line 174

::GetEDMMode

```
proc  ::GetEDMMode {  }
```

Get EDM (Electronic Distance Meter) mode

Returns:
mode

Defined in:
tca1800.tcl, line 309

::GetFace

```
proc  ::GetFace {  }
```

Get face information

Returns:

faceinfo 1/2

Defined in:

tca1800.tcl, line 296

::GetId

```
proc  ::GetId {  }
```

Read instrument id

Returns:

istrument id

Defined in:

tca1800.tcl, line 472

::GetInstrument

```
proc  ::GetInstrument {  }
```

Read instrument name

Returns:

instrument name

Defined in:

tca1800.tcl, line 462

::GetLock

```
proc  ::GetLock {  }
```

Get Lock status

Returns:

0/1 - off/on

Defined in:

tca1800.tcl, line 202

::GetPc

```
proc  ::GetPc {  }
```

Get prism constant

Returns:

pc, mm unit

Defined in:

tca1800.tcl, line 149

::GetRefCorr

```
proc  ::GetRefCorr {  }
```

Get refraction correction setting

Returns:

refraction correction as a list {on earthradius scale}

Defined in:

tca1800.tcl, line 242

::GetStation

```
proc  ::GetStation {  }
```

Get station co-ordinates (dummy proc for compatibility puspose)

Returns:

-1 (always error)

Defined in:

trimble.tcl, line 267

::Measure

```
proc  ::Measure { {prg 1} {wait 2000} {incl 0} }
```

Measure distance

Parameters:

`prg` - measure program 1/2/.. standard/track TBD
`wait` - not used (only for compability purposes)
`incl` - not used (only for compability purposes)

Returns:

error code or list of observations: {{7 hz} {8 v} {9 sd}}

Defined in:

trimble.tcl, line 356

::MoveAndMeasure

```
proc  ::MoveAndMeasure { hz v {units RAD} {atr 0} }
```

Rotate the instrument and measure distance

Parameters:

`hz` - horizontal direction
`v` - zenith angle
`units` - angle unit, optional (default RAD)
`atr` - 0/1 atr off/on

Defined in:

tca1800.tcl, line 426

::OpenCom

```
proc  ::OpenCom { {par tca1800.com} }
```

Open com port, port number and other parameters are read from a file.

Set global variable com

Parameters:

`par` - name of parameter file, optional (default tca1800.com)

Returns:

0 on success nonzero in case of error

Defined in:

tca1800.tcl, line 60

::ReadCom

```
proc  ::ReadCom {  }
```

Read input from com port

Input chars are added to global input buffer (buf)

Defined in:
tca1800.tcl, line 24

::Send

```
proc  ::Send { msg }
```

Send message to the instrument and wait for answer

Parameters:
`msg` - message to send

Returns:
return 0 on sucess or nonzero error status

Defined in:
tca1800.tcl, line 88

::SetAtmCorr

```
proc  ::SetAtmCorr { lambda pres dry wet }
```

Set atmospheric correction settings

Parameters:
`lambda` - -
`pres` - pressure value
`dry` - dry temperature
`wet` - wet temperature

Returns:
0 or error code

Defined in:
tca1800.tcl, line 233

::SetATR

```
proc  ::SetATR { atr }
```

Set ATR status on/off

Parameters:
`atr` - 0/1 = off/on

Returns:
0 or error code

Defined in:
tca1800.tcl, line 162

::SetEDMMode

```
proc  ::SetEDMMode { mode }
```

Set EDM mode

Parameters:

`mode` - EDM mode to set
1 - single to tape
2 - single to prism
3 - single fast to prism
4 - single long range
5 - single short range
6 - tracking to prism
7 - tracking dynamic
8 - tracking reflector less
9 - tracking fast
10- averaging to prism
11- averaging short range
12- averaging long range

Returns:

0 on success, non zero in case of error

Defined in:

tca1800.tcl, line 334

::SetLock

```
proc  ::SetLock { lock }
```

Set Lock status on/off

Parameters:

`lock` - 0/1 = off/on

Returns:

0 or error code

Defined in:

tca1800.tcl, line 190

::SetOri

```
proc  ::SetOri { ori {units RAD} }
```

Set orientation angle

Parameters:

`ori` - orientation angle
`unit` - unit for orientation angle, optional (default RAD)

Returns:

return 0 or error code

Defined in:

tca1800.tcl, line 346

::SetPc

```
proc  ::SetPc { pc }
```

Set prism constant

Parameters:

`pc` - prism constant to set, mm unit

Returns:

0 on succes or nonzero error code

Defined in:

tca1800.tcl, line 138

::SetRCS

```
proc  ::SetRCS { rcs }
```

Set RCS searching mode

Defined in:

tca1800.tcl, line 359

::SetRefCorr

```
proc  ::SetRefCorr { on r s }
```

Set refraction correction settings

Parameters:

`on` - 0/1 off/on
`r` - earth radius
`s` - refractice scale

Returns:

0 or error code

Defined in:

tca1800.tcl, line 257

::SetStation

```
proc  ::SetStation { e n z }
```

Set station coordinates

Parameters:

`e` - easting
`n` - northing
`z` - elevation

Returns:

0 or error code

Defined in:

tca1800.tcl, line 286

Procedure Detail

::CloseCom

```
proc  ::CloseCom {  }
```

Close communication port

Defined in:

trimble.tcl, line 47

::Coords

```
proc  ::Coords { {wait 100} {incl 0} }
```

Read coordinates from instrument

Parameters:

`wait` - (not used only for compatibility)
`incl` - (not used only for compatibility)

Returns:

coordinates from instrument : {{38 Easting} {37 Northing} {39 Height}}

Defined in:

trimble.tcl, line 391

::GetAngles

```
proc  ::GetAngles {  }
```

Read angles from instrument

Returns:

angles in radians in a list : {{7 hz} {8 v}}

Defined in:

trimble.tcl, line 405

::GetAtmCorr

```
proc  ::GetAtmCorr {  }
```

Get atmospheric correction settings

Returns:

atmospheric settings as a list {ppm pressure drytemp wettemp}

Defined in:

trimble.tcl, line 186

::GetATR

```
proc  ::GetATR {  }
```

Get ATR status -- dummy function for compatibility purposes

Returns:

-1 (always error)

Defined in:

trimble.tcl, line 167

::GetEDMMode

```
proc ::GetEDMMode { }
```

Get EDM (Electronic Distance Meter) mode

Returns:
mode

Defined in:
trimble.tcl, line 292

::GetFace

```
proc ::GetFace { }
```

Get face

Returns:
faceinfo 1/2

Defined in:
trimble.tcl, line 282

::GetId

```
proc ::GetId { }
```

Read instrument id

Returns:
instrument id or error code

Defined in:
trimble.tcl, line 424

::GetInstrument

```
proc ::GetInstrument { }
```

Read instrument name

Returns:
instrument name

Defined in:
trimble.tcl, line 417

::GetLock

```
proc ::GetLock { }
```

Get Lock status

Returns:
-1 (always error)

Defined in:
trimble.tcl, line 180

::GetPc

```
proc ::GetPc { }
```

Get prism constant

Returns:
pc in mm

Defined in:
trimble.tcl, line 148

::GetRefCorr

```
proc  ::GetRefCorr {  }
```

Get refraction correction settings

Returns:

refraction settings as a list {on earthradius scale}

Defined in:

trimble.tcl, line 233

::GetStation

```
proc  ::GetStation {  }
```

Get station co-ordinates (dummy proc for compatibility puspose)

Returns:

-1 (always error)

Defined in:

trimble.tcl, line 267

::Measure

```
proc  ::Measure { {prg 1} {wait 2000} {incl 0} }
```

Measure distance

Parameters:

`prg` - measure program 1/2/.. standard/track TBD
`wait` - not used (only for compability purposes)
`incl` - not used (only for compability purposes)

Returns:

error code or list of observations: {{7 hz} {8 v} {9 sd}}

Defined in:

trimble.tcl, line 356

::Move

```
proc  ::Move { hz v {units RAD} {atr 0} }
```

Rotate instrument to given direction

Parameters:

`hz` - horizontal direction, pseudo dms (ddd.mmss)
`v` - zenith angle, pseudo dms (ddd.mmss)
`units` - DEG/RAD/DMS
`atr` - not used, only for compability purposes

Returns:

0 or error code

Defined in:

trimble.tcl, line 336

::MoveAndMeasure

```
proc  ::MoveAndMeasure { hz v {units RAD} {atr 0} }
```

Rotate the instrument and measure distance

Parameters:

`hz` - horizontal direction
`v` - zenith angle
`atr` - not used only for compability purposes

Returns:

0 in case of error

Defined in:

trimble.tcl, line 380

::OpenCom

```
proc  ::OpenCom { {par trimble5503.com} }
```

Open communication port, port number and other parameters are read from a file
(sets global variable com)

Parameters:

`par` - name of parameter file, optional (default trimble5503.com)

Returns:

0 on success on nonzero error code

Defined in:

trimble.tcl, line 58

::ReadCom

```
proc  ::ReadCom {  }
```

Read input from com port.

Input chars are added to global input buffer (buf)

Defined in:
trimble.tcl, line 23

::Send

```
proc  ::Send { msg }
```

Send message to the instrument and wait for answer

Parameters:
`msg` - message to send

Returns:
0 on success or nonzero error status

Defined in:
trimble.tcl, line 85

::SetAtmCorr

```
proc  ::SetAtmCorr { ppm pres dry wet }
```

Set atmospheric correction setting

Parameters:
`ppm` - correction @parampres pressure value @paramd dry dry temperature
`wet` - wet temperature

Returns:
0 or error code

Defined in:
trimble.tcl, line 218

::SetATR

```
proc  ::SetATR { atr }
```

Set ATR status -- dummy function for compatibility purposes

Parameters:
`atr` - 0/1 = off/on

Returns:
-1 (always error)

Defined in:
trimble.tcl, line 161

::SetEDMMode

```
proc  ::SetEDMMode { mode }
```

Set EDM mode

Parameters:
`mode` - EDM mode to set 1 - single to tape 2 - single to prism 3 - single fast to prism 4 - single long range 5 - single short range 6 - tracking to p
reflector less 9 - tracking fast 10- averaging to prism 11- averaging short range 12- averaging long range

Returns:
0 on success, non zero in case of error

Defined in:
trimble.tcl, line 312

::SetLock

```
proc  ::SetLock { lock }
```

Set Lock status -- dummy function for compatibility purposes

Parameters:
`lock` - - 0/1 = off/on

Returns:
-1 (always error)

Defined in:
trimble.tcl, line 174

::SetOri

```
proc  ::SetOri { ori {units DEG} }
```

Set orientation angle

Parameters:

`ori` - orientation angle

Returns:

0 or error code

Defined in:

trimble.tcl, line 320

::SetPc

```
proc  ::SetPc { pc }
```

Set prism constant

Parameters:

`pc` - prism constant in mm

Returns:

0 on success or nonzero error code

Defined in:

trimble.tcl, line 137

::SetRefCorr

```
proc  ::SetRefCorr { on r s }
```

Set refraction correction on/off

Parameters:

`on` - 0/1 off/on

`r` - earth radius

`s` - refractive scale

Returns:

0 or error code

Defined in:

trimble.tcl, line 253

::SetStation

```
proc  ::SetStation { e n z }
```

Set station coordinates (dummy proc for compatibility puspose)

Parameters:

`e` - easting

`n` - northing

`z` - elevation

Returns:

-1 (always error)

Defined in:

trimble.tcl, line 276