

# **Tvorba grafického uživatelského prostředí pro skenovací mikroskopii**

Bc. Ondřej Václavík

---

Diplomová práce  
2014



Univerzita Tomáše Bati ve Zlíně  
Fakulta aplikované informatiky

---

Univerzita Tomáše Bati ve Zlíně  
Fakulta aplikované informatiky  
akademický rok: 2013/2014

## **ZADÁNÍ DIPLOMOVÉ PRÁCE**

(PROJEKTU, UMĚLECKÉHO DÍLA, UMĚLECKÉHO VÝKONU)

Jméno a příjmení:	<b>Bc. Ondřej Václavík</b>
Osobní číslo:	<b>A12331</b>
Studijní program:	<b>N3902 Inženýrská informatika</b>
Studijní obor:	<b>Bezpečnostní technologie, systémy a management</b>
Forma studia:	<b>prezenční</b>
 Téma práce:	 <b>Tvorba grafického uživatelského prostředí pro skenovací mikroskopii</b>
Téma anglicky:	<b>The Creation of a Graphical User Interface for Scanning Microscopy</b>

Zásady pro vypracování:

1. Seznamte se s programovacím prostředím MATLABu a jeho grafickou nadstavbou GUI.
2. Navrhněte a realizujte uživatelské prostředí v GUI pro ovládání mikroposuvů.
3. Zvolte vhodný způsob ukládání dat a jejich vizualizaci.
4. Ověřte funkčnost navrženého programu na vzorovém měření.
5. Proveďte měření vybraných vzorků a zhodnoťte dosažené výsledky.

Rozsah diplomové práce:

Rozsah příloh:

Forma zpracování diplomové práce: **tištěná/elektronická**

Seznam odborné literatury:

1. HAWKES, Peter W.; SPENCE, John C. H. Science of Microscopy : Volume I. 1st ed. New York : Springer, c2007, 747 s.
2. HAWKES, Peter W.; SPENCE, John C. H. Science of Microscopy : Volume II. 1st ed. New York : Springer, c2007, 751-1265 s.
3. TSUKRUK, V a Srikanth SINGAMANENI. Scanning probe microscopy of soft matter: fundamentals and practices. Weinheim: Wiley-VCH, c2012. ISBN 978-3-527-32743-0.
4. ZAPLATÍLEK, Karel a Bohuslav DOŇAR. MATLAB: tvorba uživatelských aplikací. 1. vyd. Praha, 2004, 215 s. ISBN 80-730-0133-0.
5. Users Guide. Agilent Technologies. Agilent 34401A 6 Digit Multimeter [online]. Seventh edition. Santa Clara, 2007 [cit. 2011-02-15]. Dostupné z WWW: <http://www.home.agilent.com/agilent/product.jsp?cc=CZ&lc=end&nid=-536902435.536880933&pageMode=PL>.

Vedoucí diplomové práce:

**Ing. Milan Navrátil, Ph.D.**

Ústav elektroniky a měření

Datum zadání diplomové práce:

**7. února 2014**

Termín odevzdání diplomové práce:

**27. května 2014**

Ve Zlíně dne 7. února 2014

prof. Ing. Vladimír Vašek, CSc.  
*děkan*



doc. RNDr. Vojtěch Křesálek, CSc.  
*ředitel ústavu*

### **Prohlašuji, že**

- beru na vědomí, že odevzdáním diplomové/bakalářské práce souhlasím se zveřejněním své práce podle zákona č. 111/1998 Sb. o vysokých školách a o změně a doplnění dalších zákonů (zákon o vysokých školách), ve znění pozdějších právních předpisů, bez ohledu na výsledek obhajoby;
- beru na vědomí, že diplomová/bakalářská práce bude uložena v elektronické podobě v univerzitním informačním systému dostupná k prezenčnímu nahlédnutí, že jeden výtisk diplomové/bakalářské práce bude uložen v příruční knihovně Fakulty aplikované informatiky Univerzity Tomáše Bati ve Zlíně a jeden výtisk bude uložen u vedoucího práce;
- byl/a jsem seznámen/a s tím, že na moji diplomovou/bakalářskou práci se plně vztahuje zákon č. 121/2000 Sb. o právu autorském, o právech souvisejících s právem autorským a o změně některých zákonů (autorský zákon) ve znění pozdějších právních předpisů, zejm. § 35 odst. 3;
- beru na vědomí, že podle § 60 odst. 1 autorského zákona má UTB ve Zlíně právo na uzavření licenční smlouvy o užití školního díla v rozsahu § 12 odst. 4 autorského zákona;
- beru na vědomí, že podle § 60 odst. 2 a 3 autorského zákona mohu užít své dílo – diplomovou/bakalářskou práci nebo poskytnout licenci k jejímu využití jen s předchozím písemným souhlasem Univerzity Tomáše Bati ve Zlíně, která je oprávněna v takovém případě ode mne požadovat přiměřený příspěvek na úhradu nákladů, které byly Univerzitou Tomáše Bati ve Zlíně na vytvoření díla vynaloženy (až do jejich skutečné výše);
- beru na vědomí, že pokud bylo k vypracování diplomové/bakalářské práce využito softwaru poskytnutého Univerzitou Tomáše Bati ve Zlíně nebo jinými subjekty pouze ke studijním a výzkumným účelům (tedy pouze k nekomerčnímu využití), nelze výsledky diplomové/bakalářské práce využít ke komerčním účelům;
- beru na vědomí, že pokud je výstupem diplomové/bakalářské práce jakýkoliv softwarový produkt, považují se za součást práce rovněž i zdrojové kódy, popř. soubory, ze kterých se projekt skládá. Neodevzdání této součásti může být důvodem k neobhájení práce.

### **Prohlašuji,**

- že jsem na diplomové práci pracoval samostatně a použitou literaturu jsem citoval. V případě publikace výsledků budu uveden jako spoluautor.
- že odevzdaná verze diplomové práce a verze elektronická nahraná do IS/STAG jsou totožné.

Ve Zlíně

.....  
podpis diplomanta

## **ABSTRAKT**

Tato práce se zabývá návrhem grafického uživatelského rozhraní pro skenovací mikroskopii a jeho tvorbou v programu MATLAB. Součástí návrhu je definování požadavků a volba vhodných nástrojů, funkcí a objektů pro její realizaci. Následně je provedeno testování vzhledu, propojení s kódem a ošetření možnosti vzniku chyb. Dále řeší vhodný způsobu zápisu a ukládání dat. V poslední fázi je provedena jejich vizualizace v 3D grafu a úpravy vzhledu pro naše potřeby. Funkčnost programu je ověřena kontrolním měřením vybraných vzorků.

**Klíčová slova:** MATLAB, mikroskopie skenující tunelovou sondou, grafické uživatelské rozhraní

## **ABSTRACT**

Abstrakt ve světovém jazyce

In this thesis, the design of a GUI for scanning microscopy and its MATLAB code are described. Part of the design is the definition of requirements and the choice of appropriate tools, functions and structures needed for its realization. Subsequently the testing of appearance is carried out and connections with the code and error handling are implemented. The design also comprises an appropriate method of recording and storing data. The last step is the visualization of the data in a 3D graph and several further customisations of the interface. The functionality of the program is verified via measurement of selected samples.

**Keywords:** MATLAB, scanning tunneling microscopy probe, graphical user interface

Poděkování, motto:

Tímto bych chtěl poděkovat mému konzultantovi Milanu Navrátilovi za věcné a často vyčerpávající připomínky a rady k diplomové práci i mimo ni. Za jeho ochotu pro poskytnutí materiálů a svého času.

Dále bych chtěl poděkovat Tomáši Martínkovi, který mě seznámil s měřicí technikou na škole a radil při nesnázích programování, při výběru literatury a dalších úkonech spojených s diplomovou prací.

Poděkování, motto a čestné prohlášení, že odevzdaná verze bakalářské/diplomové práce a verze elektronická, nahraná do IS/STAG jsou totožné ve znění:

Prohlašuji, že odevzdaná verze bakalářské/diplomové práce a verze elektronická nahraná do IS/STAG jsou totožné.

## OBSAH

<b>I</b>	<b>TEORETICKÁ ČÁST.....</b>	<b>10</b>
	<b>ÚVOD.....</b>	<b>11</b>
1.1	MATLAB .....	12
1.2	HISTORIE.....	12
1.3	VLASTNOSTI.....	13
1.4	PROMĚNNÉ V MATLABU .....	13
1.5	M-FILE , M-SOUBOR.....	14
1.6	KOMPONENTY .....	15
1.6.1	Toolboxy - Knihovny .....	15
1.6.2	Simulink .....	15
1.6.3	GUIDE .....	15
1.6.4	Rozhraní pro jiné programovací jazyky .....	16
1.7	OBJEKTOVĚ-ORIENTOVANÉ PROGRAMOVÁNÍ .....	16
1.8	ALTERNATIVY MATLABU .....	16
<b>2</b>	<b>GRAFICKÉ UŽIVATELSKÉ ROZHRANÍ.....</b>	<b>18</b>
2.1	HISTORIE GUI.....	18
2.1.1	Předchůdci GUI.....	18
2.1.2	Uživatelské rozhraní PARC .....	19
2.1.3	Další vývoj .....	19
2.1.4	Post-WIMP rozhraní .....	19
2.2	POROVNÁNÍ S PŘÍKAZOVÝM ŘÁDKEM (CLI).....	20
2.3	3D UŽIVATELSKÁ ROZHRANÍ U PC .....	20
2.4	JINÁ UŽIVATELSKÁ ROZHRANÍ.....	20
<b>3</b>	<b>GUIDE – NÁSTROJ PRO INTERAKTIVNÍ TVORBU GRAFICKÉHO ROZHRANÍ.....</b>	<b>21</b>
3.1	POŽADAVKY NA GUI, PRINCIPY DESIGNU.....	21
3.2	PROCES NÁVRHU VZHLEDU GUI .....	21
3.3	ESTETIZACE.....	22
3.4	PROSTŘEDÍ GUIDE .....	23
3.4.1	Grafické prvky - uiobjekty .....	23
3.4.2	Přehled uicontrol .....	24
3.4.3	Vlastnosti objektu a typy funkcí .....	29
3.4.4	Identifikátor objektů (handle) .....	30
<b>4</b>	<b>METODY ZOBRAZENÍ MORFOLOGICKÝCH PORVRCHŮ .....</b>	<b>31</b>
4.1	MIKROSKOPIE SKENUJÍCÍ SONDOU (SPM - SCANNING PROBE MICROSCOPY) .....	31
4.1.1	Skenovací tunelovací mikroskopie (STM – Scanning tunneling microscopy).....	32
<b>5</b>	<b>POUŽITÉ PŘÍSTROJE A MĚŘICÍ TECHNIKA.....</b>	<b>36</b>
5.1	SERVOMOTOR MERCURY M-110 1DG .....	36
5.1.1	Velikost jednoho kroku posuvu servomotoru .....	37

5.2	DIGITÁLNÍ MULTIMETR - HEWLETT PACKARD 34401A .....	38
5.3	MOŽNOSTI ZVÝŠENÍ PŘESNOSTI MĚŘENÍ - VYUŽITÍ P-611.3S NANO CUBE .....	39
<b>II PRAKTICKÁ ČÁST .....</b>		<b>41</b>
<b>6</b>	<b>SEZNÁMENÍ S PROGRAMOVACÍM PROSTŘEDÍM MATLAB.....</b>	<b>42</b>
6.1	NÁVRH A TVORBA JEDNODUCHÉHO GUI.....	45
6.2	GUIDE .....	45
6.3	DÁVKOVÉ SOUBORY (M-SOUBORY, M-FILES).....	46
6.4	FIGURE .....	46
<b>7</b>	<b>NÁVRH A REALIZACE GUI PRO OVLÁDÁNÍ MIKROPOSUVŮ .....</b>	<b>47</b>
7.1	OBEZNÁMENÍ O MĚŘENÍ A POUŽITÝCH ZAŘÍZENÍCH.....	47
7.1.1	Využitelnost měření .....	47
7.1.2	Zadávané hodnoty, délka kroku .....	47
7.2	NÁVRH GUI .....	48
7.2.1	Definovat požadavky na GUI.....	48
7.2.2	Návrh GUI – vhodné uicontrol objekty .....	49
7.3	VÝSLEDNÝ VZHLED GUI.....	49
7.4	JEDNOTLIVÉ ČÁSTI PROGRAMU .....	50
7.4.1	Měřicí část programu .....	50
7.4.2	Inicializační část programu .....	50
7.4.3	Průběh měření .....	50
7.5	SCHÉMA ALGORITMU PROGRAMU .....	52
7.6	TEST DESIGNU .....	53
7.7	ZÁPIS KÓDU A PROPOJENÍ S GUI .....	53
7.8	TEST KÓDU A OŠETŘENÍ PROTI CHYBÁM .....	53
<b>8</b>	<b>VOLBA VHODNÉHO ZPŮSOBU UKLÁDÁNÍ DAT A JEJICH VIZUALIZACE.....</b>	<b>55</b>
8.1	ZPŮSOB UKLÁDÁNÍ DAT .....	55
8.2	VIZUALIZACE DAT .....	56
8.3	PŘÍKLAD VIZUALIZACE JIŽ DŘÍVE NAMĚŘENÝCH DAT .....	57
<b>9</b>	<b>OVĚŘENÍ FUNKČNOSTI NAVRŽENÉHO PROGRAMU .....</b>	<b>58</b>
9.1	VZNIKLÉ PROBLÉMY A ŘEŠENÍ – PROGRAMOVÁ ČÁST.....	58
9.2	VZNIKLÉ PROBLÉMY A ŘEŠENÍ – MĚŘICÍ ČÁST .....	58
9.2.1	Chyby měření .....	58
9.2.2	Odhad doby měření .....	59
<b>10</b>	<b>MĚŘENÍ VYBRANÝCH VZORKŮ A HODNOCENÍ VÝSLEDKŮ.....</b>	<b>61</b>
10.1	SKENOVANÝ VZOREK - KLÍČ.....	61
10.2	SKENOVANÝ VZOREK - PĚTIKORUNA .....	63
10.3	SKENOVANÝ VZOREK - DNO NÁBOJNICE .....	65
<b>ZÁVĚR .....</b>		<b>68</b>
<b>SEZNAM POUŽITÉ LITERATURY.....</b>		<b>70</b>
<b>SEZNAM POUŽITÝCH SYMBOLŮ A ZKRATEK.....</b>		<b>73</b>
<b>SEZNAM OBRÁZKŮ .....</b>		<b>74</b>



SEZNAM TABULEK.....	76
SEZNAM PŘÍLOH.....	77

## **I.      TEORETICKÁ ČÁST**

## ÚVOD

Tato práce se zabývá návrhem a realizací grafického uživatelského rozhraní (známe pod zkratkou GUI, anglicky Graphical User Interface) pro skenovací mikroskopii. Vytvořením grafického uživatelského rozhraní tak bude usnadněno měření ostatním studentům, docentům, doktorandům, profesorům a ostatním pedagogickým, výzkumným, vědeckotechnickým pracovníkům. Uživatel tak nebude muset mít znalosti programování a bude schopen jednoduše proměřit potřebné vzorky. Program musí být ošetřen proti chybám ze strany uživatele (chybné hodnoty, špatný rozsah a podobně), aby nedošlo k poškození přístrojů. Program musí být jednoduchý, intuitivní na ovládání, ale musí zároveň obsahovat všechny potřebné funkce.

V teoretické části je přiblíženo, co je MATLAB (zkratka slov Matrix Laboratory), jeho historie, vlastnosti a jeho popis, co je grafické uživatelské rozhraní, jak se v průběhu času vyvíjelo, programovací prostředí a programovací jazyk MATLAB a jeho grafická nástavba GUIDE (anglicky Graphical User Interface Development Environment). Dále je popsána skenovací mikroskopie, zejména pak mikroskopie skenující tunelovou sondou. V poslední kapitole jsou zmíněny měřicí přístroje a zařízení a sestavy zařízení, na kterých je prováděno měření.

V praktické části je popsáno programovací prostředí MATLAB, definování požadavků na GUI, návrh a realizace grafického uživatelského rozhraní a jeho testování. Dále je přiblížen program pro měření mikroposuvů, jeho úpravy a propojení s GUI. Podstatnou částí je vyladění chyb uživatelského rozhraní, měřicího programu i chyb měření. Součástí praktické části je i kontrola funkčnosti programu a ukázka programu na konkrétních měřených vzorcích. V poslední kapitole jsou popsány způsoby ukládání, vizualizace a prezentace naměřených dat.

## 1.1 MATLAB

MATLAB (zkratka slov matrix laboratory, volně přeloženo „maticová laboratoř“) je interaktivní programové prostředí a skriptovací programovací jazyk čtvrté generace. Program MATLAB je vyvíjen společností MathWorks a v březnu 2014 vyšla zatím poslední verze R2014a, která je k dispozici pro operační systémy Linux (32-bit, 64-bit), Windows (32-bit, 64-bit), Mac OS X (64-bit). MATLAB umožňuje počítání s maticemi, vykreslování 2D i 3D grafů funkcí, implementaci algoritmů, počítačovou simulaci, analýzu a prezentaci dat i vytváření aplikací včetně uživatelského rozhraní. Původně byl jazyk určen pro matematické účely, ale časem byl upraven, byly přidány nové funkce a rozšíření, rozrostl se různými směry a dnes je využitelný v široké paletě aplikací. V roce 2004 měl MATLAB přes milión uživatelů a to především z řad vědeckotechnických pracovníků, studentů a zaměstnanců vysokých škol. MATLAB je využíván pro vědecké a výzkumné účely a to jak v soukromém sektoru, tak i v akademických řadách. Hlavní oblastí využití jsou technické obory a ekonomie. Někteří odborníci nepovažují MATLAB za programovací jazyk, jiní o něm zase říkají, že je velice cenným a užitečným programovacím jazykem. Klíčovou datovou strukturou při výpočtech v MATLABu jsou matice. Vlastní programovací jazyk vychází z jazyka Fortran. [1][2][3]

## 1.2 Historie

MATLAB vytvořil na konci sedmdesátých let profesor Cleve Moler, který v té době působil na Univerzitě v Novém Mexiku na katedře informačních technologií. Navrhnul MATLAB z důvodu, aby jeho studenti mohli využívat knihovny LINPACK a EISPACK bez nutnosti se učit programovací jazyk Fortran, který se mnoho let využíval jako hlavní jazyk pro matematické výpočty. Programování ve Fortranu bylo složité a skrývalo mnoho nástrah při řešení matematických výpočtů. MATLAB se velice rychle rozšířil i na další univerzity a získal si veliké množství uživatelů a fanoušků. Široké využití našel především v aplikované matematice. V roce 1983, když byl Cleve Moler na návštěvě na Stanfordově univerzitě, se o MATLAB začal zajímat Jack Little, který v softwaru uviděl ekonomický potenciál. Do této doby byl MATLAB zdarma. Jack Little přepsal MATLAB do jazyka C, přidal některé další funkce a knihovny a v roce 1984 založili Little, Moler a Steve Bergert společnost MathWorks, která dále pokračovala ve vývoji a nabídla produkt na trh.

První verze pro PC XT byla vydána kolem roku 1985. Základním problémem bylo nedostatek paměti a z toho plynoucí omezení na maximální velikost matic. Po příchodu PC

AT rychle následovala verze MATLABu pro tento počítač. Zde byla maximální velikost matice omezena fyzickou velikostí paměti (max 16 MB). Vzhledem k tehdejším cenám paměti však nebylo osazení větší paměti běžné. Velké obliby doznala verze MATLAB 386, která byla určena pro PC s 32bitovým procesorem Intel 80386, která využívala virtuální paměť. Program pracující s virtuální pamětí umožňuje využívat více operační paměti, než je dostupná fyzická paměť RAM, protože dochází k odkládání dat na pevný disk počítače. Tato skutečnost sice vede k zpomalení výpočtů, ale je možné výpočty na velkých maticích vůbec provést. [1][2][3]

### 1.3 Vlastnosti

Programovací jazyk MATLAB je integrované prostředí, které je určené pro vědeckotechnické účely, simulace, paralelní výpočty apod. Zahrnuje výpočty, vizualizaci a programování do uživatelsky ovladatelného prostředí. Problémy a řešení jsou nejčastěji vyjádřeny pomocí známých matematických vztahů. Typické oblasti použití:

- inženýrské výpočty,
- tvorba algoritmů,
- modelování a simulace,
- analýza dat,
- vědecká a inženýrská grafika,
- tvorba aplikací (včetně grafického rozhraní).

Mezi základní vlastnosti lze zahrnout vlastnost, že veškeré objekty v MATLABu jsou považovány za prvky pole (matice). Tyto prvky však mohou být nejen čísla, proměnné, ale i složitější struktury jako například obrázky. Výkonnost MATLABu je rozšiřována díky navazujícímu softwaru, které tvoří především soubory programu tzv. "toolboxy", orientované zpravidla na daný problém nebo uživatelem sestavené programy, tzv. m-files (m-soubory). [2][3]

### 1.4 Proměnné v MATLABu

MATLAB má slabou dynamickou typovou kontrolu. To znamená, že proměnné v MATLABu nemají po deklaraci určený datový typ a mění datový typ během své existence. Je možné do jedné proměnné uložit datový typ integer a následně v kódu do té samé

proměnné uložit textový řetězec, kterým přepíšeme původní hodnotu. Dynamické typování je pružnější a mnohdy pohodlnější pro programátora, ovšem je daleko náchylnější ke vzniku chyb. Programátor si typovou kontrolu musí hlídat sám a některé takto vzniklé chyby se projeví až daleko od místa svého vzniku a jsou tak jen těžko odhalitelné.

- Proměnné v MATLABu nevyžadují deklaraci, vzniknou prvním přiřazením hodnoty.
- Čtení z neexistující proměnné vyvolá chybu.
- Změny typu a velikosti proměnných probíhají automaticky.

Základním druhem proměnné je matice. Každá matice může v MATLABu obsahovat:

- Čísla v různých formátech (celočíslná, s plovoucí desetinnou čárkou, komplexní),
- znaky (s vektorem znaků je zacházeno jako s řetězcem),
- struktury, které mohou obsahovat další matice nebo struktury,
- symbolické proměnné nebo speciální typy (např. přenosová funkce 'tf').

Matice může mít prakticky libovolný počet rozměrů, v každém rozměru jsou prvky označeny celými kladnými čísly od 1 do  $n$ . Matice o rozměrech  $1 \times 1$  je označována jako skalár, matice  $1 \times n$  nebo  $m \times 1$  je označována jako vektor. Matice jsou vždy obdélníkové (nejsou možná pole polí jako např. v C++ s různými délkami druhé úrovně). Všechny prvky jedné matice musí být stejného typu ('double', 'int', 'logical', 'struct', 'sym', ...).

Omezení matic na stejný typ ve všech prvcích odstraňují později zavedení proměnné 'cell', k jejímž prvkům se přistupuje pomocí složených závorek. Jednotlivé prvky 'cell' mohou mít libovolný obsah. [3][4]

## 1.5 M-file , m-soubor

Soubory s koncovkou \*.m, takzvané m-files nebo m-soubory, obsahují definice funkcí, skriptů nebo tříd. Sami si můžete m-soubory vytvářet a tak přidávat do MATLABu nové funkce a rozšiřovat tak jeho využití. Název funkce v m-file musí být stejný jako název souboru. U funkce si nastavíte vstupní parametry a návratové hodnoty, následný komentář, slouží jako nápověda, popisující k čemu daná funkce slouží.

MATLAB podporuje funkce s různým počtem vstupních i výstupních parametrů. Veškerý text na řádce za znakem '%' je brán jako komentář. Středníky v MATLABu neslouží k ukončení řádku nebo příkazu, ale k zabránění výpisu na obrazovku. Kdybychom středníky nepoužili, na obrazovku by se vypsaly všechny kroky a jejich mezivýsledky, které funkce provádí. [5]

## **1.6 Komponenty**

### **1.6.1 Toolboxy - Knihovny**

Důležitou částí instalace MATLABu jsou knihovny funkcí (adresáře s m a mex soubory), které jsou nazývány toolboxy. Toolboxy obsahují vždy uceleným způsobem, včetně dokumentace a příkladů, zpracovaný určitý obor numerické matematiky, analytické matematiky, statistiky, systémového přístupu k regulacím a další obory, ve kterých nachází MATLAB uplatnění. Takovýchto balíků je dnes k dispozici asi 35. [6]

### **1.6.2 Simulink**

Simulink je program, který využívá MATLAB a jeho funkce k simulaci dynamických systémů. Je mladší než MATLAB. Jeho první verze byla k dispozici pro MATLAB 4. Simulink má trochu jiné uživatelské rozhraní než MATLAB. Zatímco u MATLABu je stále nejdůležitější příkazový řádek, ovládání Simulinku je jednodušší a intuitivnější, ale pokročilejší funkce nelze provádět bez znalosti jazyka MATLAB. Interaktivní způsob tvorby a simulace modelů se spouští z příkazové řádky systému MATLAB příkazem simulink. Po spuštění je vytvořeno okno pro tvorbu nového modelu a okno obsahující základní nabídku otvírání knihoven zdrojů signálů, základních spojitých, diskrétních a nelineárních bloků a bloků pro zobrazování a ukládání signálů. Pod touto interaktivní obálkou se skrývá systém velmi podobný grafickému subsystému s obdobnými funkcemi simget a simset. Další vrstva funkcí umožňuje již komfortnější neinteraktivní tvorbu modelů systémů. Pro obvyklého uživatele však není nutné o implementaci a programování modelů přemýšlet. [7]

### **1.6.3 GUIDE**

Prostředí, které umožňuje vytvářet aplikace s grafickým rozhraním, se nazývá GUIDE (Graphical User Interface Development Environment). Obsahuje následující vlastnosti:

- umožňuje vytvářet a editovat uživatelské rozhraní pomocí základních komponent (checkbox, sliders, tables apod.)
- všechny komponenty, které jsou vytvořeny v tomto prostředí, lze měnit za běhu aplikace
- vzhled vytvořené GUI aplikace je ukládán do souboru s příponou \*.fig a jeho zdrojový kód s příponou \*.m

Spuštění průvodce pro tvorbu GUI aplikací je možné více způsoby. Jedním z nich je využít základní menu File/New/GUI nebo zápisem a potvrzením příkazu GUIDE v hlavním prostředí MATLABu. [8]

#### 1.6.4 Rozhraní pro jiné programovací jazyky

MATLAB Builder je součástí MATLABu verze R2006b a vyšší. Společně s MATLAB kompilátorem umožňuje vytvářet Java komponenty (jar archívy) z programů MATLAB. Java komponentu lze pak nasadit např. do desktopové aplikace nebo na webový server.

### 1.7 Objektově-orientované programování

MATLAB podporuje objektově-orientované programování a podle stránek MathWorks vám umožní vyvíjet komplexní aplikace rychleji než v jiných programovacích jazycích jako je Java nebo C#. V MATLABu můžete vytvářet třídy, používat dědičnost, nastavovat události a posluchače pro pozorování objektů nebo využívat návrhových vzorů používaných i v jiných objektově-orientovaných jazycích a přitom zůstat ve vysokoúrovňovém jazyce. Objektově orientované programování významně zjednodušuje a zpřehledňuje tvorbu složitějších aplikací. [9]

### 1.8 Alternativy MATLABu

Na trhu se vyskytuje velké množství konkurentů pro MATLAB. Z komerčních produktů to jsou například Mathematica, Maple, IDL od společnosti ITT Visual Information Solutions nebo také Metlynx. Existují také open source alternativy k MATLABU, jako je GNU Octave, FreeMat a Scilab, které jsou s jazykem MATLAB v leccem srovnatelné ovšem kvality prostředí MATLABu zdaleka nedosahují. Existují také různé knihovny, které přidávají podobnou funkčnost jako má MATLAB do jiných



existujících jazyků. Takovéto knihovny jsou například IT++ pro C++, Perl Data Language pro Perl nebo SciPy společně s NumPy a Matplotlib pro Python. [10]

## 2 GRAFICKÉ UŽIVATELSKÉ ROZHRAŇÍ

Grafické uživatelské rozhraní (anglicky Graphical User Interface, známe pod zkratkou GUI) je uživatelské rozhraní, které umožňuje ovládat počítač pomocí interaktivních grafických ovládacích prvků. Na monitoru počítače jsou zobrazena okna, ve kterých programy zobrazují svůj výstup. Uživatel používá klávesnici, myš a grafické vstupní prvky jako jsou menu, ikony, tlačítka, posuvníky, formuláře a podobně.

GUI lze použít v počítačích, přenosných zařízeních, jako jsou přehrávače MP3, přenosné přehrávače médií a herní zařízení, domácích spotřebiče, kancelářské a průmyslové vybavení, a jiné. GUI představuje informace a akce, které jsou pro uživatele zobrazovány pomocí grafických ikon a vizuálních indikátorů, což je rozdíl oproti textovým rozhraním (CLI) nebo textové navigace. Akce jsou obvykle prováděny prostřednictvím přímé manipulace s grafickými prvky. Termín GUI je omezena na rozsah dvojrozměrných obrazovek se schopností popsat generické informace. Uživatelskému rozhraní se věnuje zejména výzkum počítačové vědy na PARC (Palo Alto Research Center). Termín GUI není obvykle aplikován na jiné typy rozhraní s nízkým rozlišením, které nejsou generické, jako například videohry, kde se dává přednost termínu HUD (Head-Up-Display). [4][5]

### 2.1 Historie GUI

První grafické uživatelské rozhraní (WIMP) bylo vyvinuto v roce 1973 ve vývojových laboratořích společnosti Xerox. Oblibu mezi uživateli získalo spolu s počítači Mac kolem roku 1984 a následně i v Microsoft Windows. [4]

#### 2.1.1 Předchůdci GUI

Jeden z předchůdců GUI byl vyvinut v Stanford Research Institute. Práce vedl Douglas Engelbart. V prostředí byly odkazy, se kterými se manipulovalo pomocí myši. Tento koncept odkazů byl dále vylepšován a rozšířen pracovníky Xerox PARC, zejména Alanem Kayem. GUI bylo primární rozhraní pro počítače Xerox Alto. Mnoho dalších moderních a univerzálních GUI bylo odvozeno z tohoto systému.

Ivan Sutherland vyvinul v roce 1963 systém „Sketchpad“. Bylo používáno pero pro vytvoření objektů a manipulaci s nimi v technických výkresech. [4][5]

### 2.1.2 Uživatelské rozhraní PARC

Uživatelské rozhraní PARC se skládalo z grafických prvků, jako například oken, nabídek (menu), „radio“ polí, zatrhávacích tlačítek a ikon. Toto rozhraní začalo používat spolu s klávesnicí také polohovací zařízení. Tyto aspekty byly zdůrazněny používáním alternativního názvu WIMP, který je zkratkou pro názvy windows (okna), icons (ikony), menus (nabídky) a pointing device (polohovací zařízení). [4][5]

### 2.1.3 Další vývoj

Následovatel PARCu, počítač, který měl v roce 1981 prvně GUI centralizovaně, byl Xerox 8010 Star Information systém. Následovaly počítače Apple Lisa (1983), Apple Macintosh 128K(1984), dále Atari ST a nakonec Commodore Amiga (1985).

Rané GUI příkazy, před příchodem IBM Common User Access, používaly různé příkazové sekvence pro různé programy. Například klávesa F3 aktivovala nápovědu v programu WordPerfect. Nabídky (menu) byly přístupné pomocí různých kláves (control v WordStar, Alt nebo F10 v programech společnosti Microsoft, pomocí "/" v Lotusu 1-2-3, F9 v Norton Commanderu).

Kvůli těmto programovým rozdílům byly vyráběny plastové nebo dřevěné masky, které byly na plochách kolem kláves. Na nich byly napsány funkce platné pro různé programy. [4]

### 2.1.4 Post-WIMP rozhraní

Aplikace na menších přenosných zařízeních (například chytré mobilní telefony nebo tablety), pro které WIMP není uzpůsoben nejlépe, používají novější techniky interakce s uživatelem. Tyto techniky jsou obecně nazývány Post-WIMP.

Začátkem desátých let 21. stol. se začala ve větší míře objevovat zařízení založená na ovládání dotyky (Android nebo iPhone), jejichž uživatelská rozhraní jsou Post-WIMP. Uživatelé používají více než jeden prst pro interakci se zařízením prostřednictvím displeje. To jim dovolí provádět akce typu přibližování (pomocí sbíhání dvou prstů) nebo rotace s objekty zobrazenými na displeji (kroužení dvěma prsty). Je důležité podotknout, že by se takové akce neprovedly pouze s využitím myši. [4][5]

## 2.2 Porovnání s příkazovým řádkem (CLI)

GUI vzniklo jako potřeba nahradit rozhraní příkazového řádku něčím, co by se lidé rychleji naučili a všechny příkazy nemuseli psát. Příkazový řádek povoluje velkou efektivitu, pokud se uživatel naučí příkazy, ale naučení zabere nějaký čas. Další věcí je ta, že používání příkazového řádku může být pomalé, když uživatel zadává příkazy s mnoha parametry s/nebo cesty k souborům na disku. WIMP poskytuje různé tlačítka apod. reprezentující rozličné systémové příkazy, a to na několik kliknutí.

Na druhou stranu, GUI může být horší tím, že některá nastavení jsou příliš hluboko v systému. Ale klikání ve WIMP může být snadnější v porovnání s tím, že v příkazových řádcích neplatí všechny příkazy pro všechny adresáře nebo prostředí. Často je proto nutné přeskakovat z adresáře do adresáře.

Většina moderních operačních systémů poskytuje jak WIMP, tak i CLI, ačkoli GUI získává u běžných uživatelů větší pozornost. Aplikace mohou mít obě rozhraní. Často je to tak, že první bylo CLI, a GUI je pouze jejím obalem, zjednodušujícím některé funkce. To je proto, aby vývojáři mohli vyzkoušet funkčnost programu, aniž by se obtěžovali s vývojem grafického rozhraní. To je povětšinou v operačních systémech typu UNIX. Umožňuje to uživatelům používat programy jako automatizované skripty, nejen pouze GUI. [4]

## 2.3 3D uživatelská rozhraní u PC

Označení 3D není přesné, protože počítačové obrazovky jsou ve skutečnosti pouze dvoudimenzionální. Nicméně, různá grafická prostředí používají tři rozměry. Výšku a šířku doplňují vrstvením nebo stohováním objektů na sebe. Bývá to doprovázeno průhledností objektů. 3D našlo své uplatnění ve filmové produkci. Samozřejmostí je, že tím, jak narůstá složitost výpočtu 3D animací, musí také narůstat výkon hardwaru, na kterém výpočty probíhají. [4]

## 2.4 Jiná uživatelská rozhraní

Kromě grafických existují i jiná uživatelská rozhraní:

- textové uživatelské rozhraní (s menu, tlačítka a myši)
- příkazový řádek (příkazy se zadávají jejich zapsáním pomocí klávesnice)
- braillovský řádek, hlasová rozhraní a další [4]

### 3 GUIDE – NÁSTROJ PRO INTERAKTIVNÍ TVORBU GRAFICKÉHO ROZHRAŇÍ

GUIDE (anglicky Graphical User Interface Development Environment) je vývojové prostředí pro grafické uživatelské rozhraní. Je to objektově orientované programovací prostředí, které usnadňuje tvorbu GUI.

#### 3.1 Požadavky na GUI, principy designu

Při tvorbě grafického uživatelského rozhraní je třeba mít stále na mysli základní poslání GUI – ulehčení ovládání aplikace uživatelem. Jednoduchost (robustnost), jasnost, elegance, přímost jsou základní vlastnosti pro každé GUI. Pohyb ve vaší aplikaci by měl být jednoduchý a rychlý. Kliknutí na ikonku, tlačítko či přepínač je vždy rychlé a snadné. Další synonyma mohou být jednota, jasnost, elegance, přímocíarost.

- Provázanost (vždy musí být zajištěný návrat)

Uživatel (ať už jsme to někdy později my sami, nebo někdo zcela jiný) naší aplikace se nesmí při řízení jejího běhu ztratit ve slepé uličce bez návratu, tedy:

- Uživateli by mělo být jasné, kde se právě nachází
- Uživatel by měl vědět jaký je další krok
- Komplexnost (ošetření všech eventualit)
- Přehlednost, přívětivost, pohodlnost

Všechny objekty musí být pro uživatele srozumitelné, přehledné, předávající mu potřebné informace o programu.

- Intuitivnost

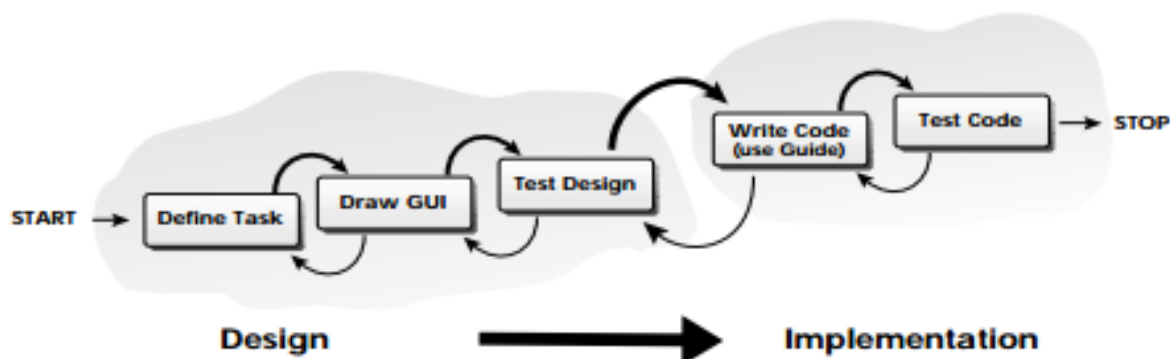
Je velmi důležité při tvorbě GUI. Ovlivňuje vnímání uživatele, pomáhá porozumění, pochopení programu. Je-li program intuitivní, tak uživatel nepotřebuje manuál pro jeho ovládání. [5][6][7][8]

#### 3.2 Proces návrhu vzhledu GUI

Proces návrhu vzhledu probíhá v několika krocích. Je důležité dokončit daný krok před tím, než začneme provádět další. Obecně platí, že nezačínáte s zápisem kódu (s

programováním) dokud není hotový design. Jinak vám bude dlouho trvat, než budete spokojení s finálním vzhledem. Návrh vzhledu prochází těmito kroky(viz obrázek 1) :

- Definovat problém
- Nakreslit GUI
- Testovat design
- Zapsat kód
- Otestovat kód



Obrázek 1 Proces návrhu vzhledu GUI [8]

[8]

### 3.3 Estetizace

Významný posuv v tomto oboru jsme zaznamenali v roce 1996, vznikl trend „Estetizace informačních nástrojů“. Vznikl v závislosti na konkurenci produktů, skrze neuspokojované potřeby zákazníka a v závislosti na posuvu v oblasti sociální a ekonomické. Vývojáři si uvědomili, že hodnotu jejich produktu znatelně zvyšuje grafické uživatelské prostředí.

Kritérium pro správný design se stala přátelskost, hravost, jemnost, expresivnost, módnost, kulturní identičnost, estetika, emoce. Proto se změnil i moderní design „forma následuje funkci“ byl nahrazen novým „forma následuje emoci.“ [9]

### 3.4 Prostředí GUIDE

Průvodce GUI (GUIDE) je grafické interaktivní vývojové prostředí, obsahující všechny grafické objekty typu uicontrol, kterým je možno ovládat běh aplikace. Generuje automaticky základní zdrojový kód pro ovládání. Vzhled vytvořeného GUI ukládá GUIDE do souboru s příponou \*.fig a jeho zdrojový kód do souboru \*.m

Vyvolání průvodce je možné více způsoby. Jedním z nich je spuštění dvojklikem levého tlačítka myši z okna Launch Pad, nebo zápisem a potvrzením příkazu „GUIDE“ v hlavním prostředí MATLABu (Command Window). Třetí možností je využít základní menu File/New/GUI.

Průvodce má podobu standardního okna s řadou rozbalovacích menu, ze kterých jsou přístupné všechny možnosti a nastavení. Pruh ikon, které jsou umístěny na hlavním panelu, obvykle zastupují některé důležité a často používané povely.

Na levé straně, v tzv. Component Palette, jsou všechny objekty uicontrol. Pracovní plochu tvoří čtvercová síť – Layout Area, umožňující hrubě vizuálně určovat souřadnice jednotlivých objektů umístěvaných na plochu. V pravém dolním rohu je malý černý čtvereček. Ten dovoluje pomocí levého tlačítka myši (kliknutím a držením) přizpůsobit rozměry čtvercové sítě Layout Area dle našich potřeb.

První dvě vodorovné sekce ikon (obsahující 3 a 5 položek) jsou standardní a dobře známé další tři sekce jsou specifické. Najetím myši na jednotlivé ikonky se standardně ve žlutém políčku zobrazí jejich funkce. Všechny tyto nové ikonky jsou vybarvené a tudíž aktivní. Devátá ikonka zprava je Alignment Tool – nástroj pro zarovnávání grafických objektů. [6]

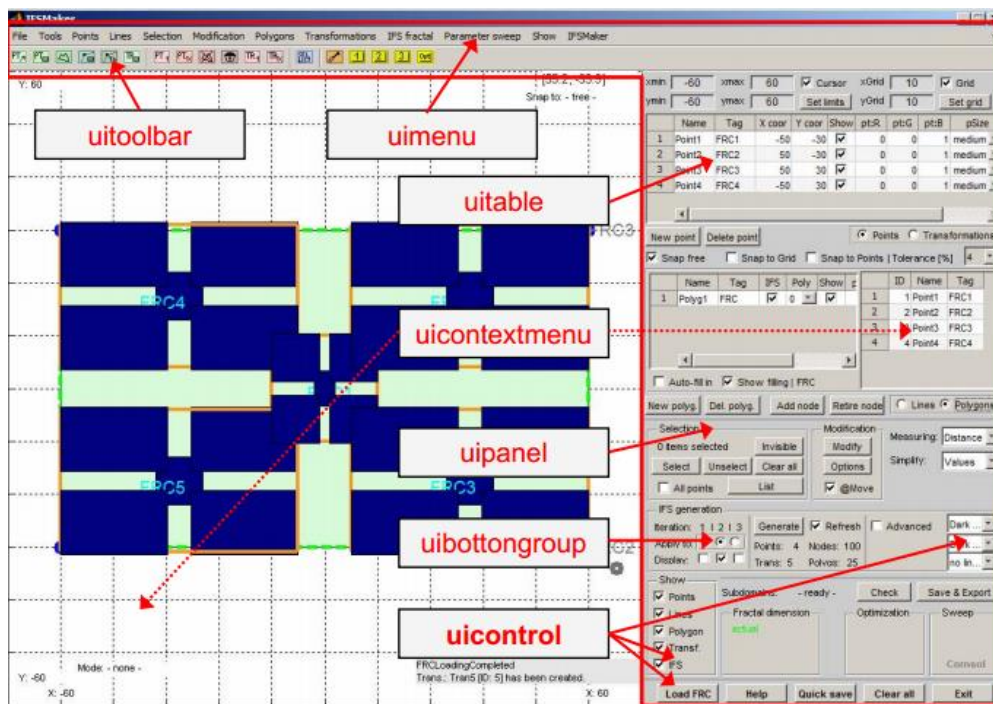
#### 3.4.1 Grafické prvky - uiobjekty

GUI v programu MATLAB je tvořeno uiobjekty (viz obrázek).

- Uitoolbar
- Uimenu
- Uitable
- Uicontextmenu
- Uipanel

- Uibottongroup
- Uicontrol

Pro lepší představu jsou zde zakresleny na obrázku. [10]



Obrázek 2 Přehled ui-objektů [10]

### 3.4.2 Přehled uicontrol

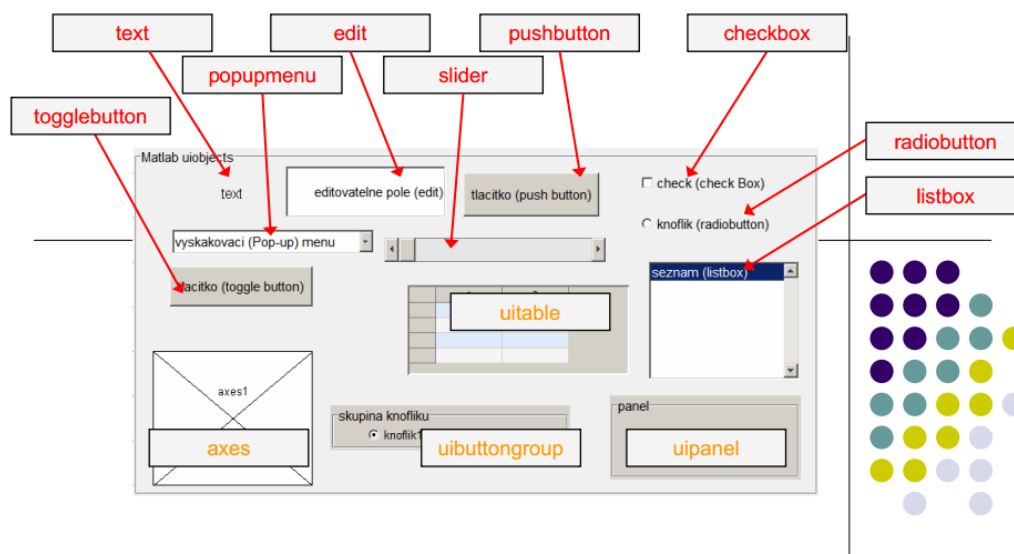
Do uicontrol patří tyto prvky:

Tooglebutton, text, popupmenu, edit, slider, pushbutton, checkbox, radiobutton a listbox.

Zde jsou znázorněny graficky (viz obrázek 3) :

[10]





Obrázek 3 Přehled uicontrol prvků [10]

## FIGURE

Je okno obsahující GUI ať už naprogramované nebo vytvořené průvodcem. V okně Figure jsou zobrazeny všechny již zmíněné uicontrol objekty. Objekt Figure vytváří samostatné grafické okno. Všechny event. okna jsou potomkem objektu root. Všechny podružné grafické objekty jsou potomkem objektu Figure. a jsou v daném okně zobrazeny. Voláme-li podružnou funkci bez existence okna, vytvoří se nové. Objekt Figure obsahuje veškeré již zmíněné objekty v grafické podobě.

## AXES, AXIS

Definují oblast v grafickém okně Figure, kam jsou umísťovány potomci objektu axes. Nejčastější využití axes jako os grafu/obrázku (2D i 3D). Všechny podružné objekty k objektu axes generují osy i pokud zatím neexistují (podobně jako u Figure). Umožňují zobrazit grafy. Není to objekt uicontrol, ale nezbytný v tomto výčtu, neboť je v mnoha případech výstupem ovládaným pomocí GUI.

## PUSH BUTTON

Jednostavové tlačítko. Akce se generuje jeho stlačením, čímž se změní navenek jeho vzhled. Uvolněním se vrací do původní nestlačené polohy.

String – nápis na tlačítku

Tag – jméno subfunkce v aplikačním m-souboru

## TOGGLE BUTTON

Dvoustavové tlačítko – přepínač. „Pamatuje si“ poslední polohu. Callback rutina musí uchovat aktuální polohu tlačítka. MATLAB nastavuje hodnotu (Value) rovnu buď Max, je-li přepínač stlačen (implicitně 1), nebo Min, je-li přepínač v nesepnuté poloze (implicitně 0).

## RADIO BUTTON

Podobá se funkcí zaškrťovacímu políčku (Check Box). Radiobutton má dva stavy „vybrán – nevybrán“, což je uchováno ve vlastní Value „Max – Min“. Každému radiobuttonu musí být přidána příslušná subfunkce. Argumentem je matice obsahující handle ostatních radiobuttonů ve skupině, které musí být „nevybrán“.

## CHECK BOX

Zaškrťovací políčko generuje akci, je-li vybráno a tento svůj stav uchovává ve vlastnosti Value „Max – Min“ (1 – 0). Určení momentálního stavu se děje dotazem na Value.

## EDIT TEXT

Umožňuje uživateli vkládat nebo modifikovat textový řetězec. Vlastnost String obsahuje text vložený uživatelem.

## STATIC TEXT

Užívá se nejčastěji jako název k uicontrol objektům nebo ke sdělení informací.

## SLIDER

Používá se numerický vstup v určitém rozsahu, čímž umožňuje uživateli pohybovat jezdcem. Pohyb se děje plynule tažením jezdce myši nebo skokově kliknutím do dráhy jezdce. Pozice jezdce je indikována numerickou hodnotou pozice posuvníku

Value - Obsahuje numerickou hodnotu pozice posuvníku. Nebo může být tato pozice nastavena a ovládána z jiného místa programu. Pak musí zpětná vazba obsahovat příkaz:

```
Slider_value = get(handles.slider1, 'Value');
```

Max – definuje maximální hodnotu vlastnosti Value

Min – Definuje minimální hodnotu Value

Slider step – Řídí jemnost změny hodnoty vlastnosti Value. Je specifikovaná vektorem se dvěma hodnotami. „Defaultní“ nastavení [0.01 0.10] znamená 1% krok při tažení jezdce myši a 10% krok při stisknutí myši do dráhy jezdce. Aktuální hodnoty jsou závislé na rozsahu Min – Max.

## FRAME

Rámy jsou obdélníkové oblasti zahrnující části okna Figure. Používají se k vizuálnímu ohraničení určitých skupin objektů uicontrol s výjimkou Axes. Nemají zpětnovazební volání. Jsou neprůhledné, objekty se umísťují na ně. Vizuálně sdružují vždy jeden Slider s jedním polem Edit Text. Přetáhneme-li objekty na Frame, jsou neviditelné (jsou umístěny pod Frame). Kliknutím pravého tlačítka na Frame a vybráním volby Send to Back se Frame umístí pod objekty a tyto jsou viditelné. Bring to front má opačný účinek.

## LIST BOX

Zobrazuje seznam, ze kterého je možno volit jednu nebo více položek. Listbox rozlišuje mezi jednoduchým kliknutím a dvojklikem na položku a nastavuje podle toho vlastnosti obrázku na normal nebo open. Po kliknutí se šipku zobrazuje seznam, ze kterého je možno volit jen jednu položku. Tím se trochu podobá sérii radiobuttonů.

Value – je vektor obsahující indexy řetězců vybraných položek. Implicitně je nastaveno zvýraznění položky, je-li Listbox zobrazen poprvé. Je-li třeba aby nebyla zvýrazněna žádná položka, pak nutno nastavit Value jako prázdnou, [ ].

Vlastnost ListboxTop definuje, která položka seznamu je nastavena jako horní, je-li položek tolik, že nemohou být zobrazeny všechny najednou. ListboxTop je index v matici řetězců definovaný vlastností String. Jeho hodnota musí být mezi jedničkou a počtem řetězců. Není-li hodnota celočíselná, pak je rovna nejbližšímu nižšímu číslu. Hodnoty vlastností Min a Max určují, zda je možno vybrat více položek, nebo jen jednu.

Je-li  $\text{Max} - \text{Min} > 1$ , pak je možný výběr více položek.

Je-li  $\text{Max} - \text{Min} \leq 1$ , není výběr více položek možný.

Listbox rozlišuje mezi jednoduchým kliknutím a dvojklikem na položku a nastavuje podle toho vlastnost obrázku SelectionType na normal nebo open. MATLAB vyhodnocuje zpětnou vazbu Listboxu po uvolnění tlačítka myši, nebo po uvolnění klávesy (včetně šipek), které mění vlastnost Value. To znamená, že příkazy zpětné vazby se vykonají po prvním kliknutí dvojkliku na jednotlivé položce. Je-li možný výběr více položek, je třeba vykonání odložit. V této situaci je třeba přidat další komponentu – Push button Hotovo (Done) a naprogramovat její zpětnovazební rutinu k dotazu na Value Listboxu, namísto tvorby zpětné vazby pro Listbox.

Je-li aplikační m-soubor vygenerován automaticky průvodcem GUIDE, je třeba buď nastavit Callback jako prázdný řetězec (‘’) a odstranit Callback subfunkci z m-souboru, nebo nechat základ Callback subfunkce, s tím, že se vykoná hned po kliknutí na položku.

## POPUP MENU

Vlastnost String zobrazuje seznam řetězců, které se v nabídce zobrazují. Vlastnost Value obsahuje index odpovídající řetězci vybrané položky. První položka v seznamu nabídek má index 1. Není-li popup menu otevřeno, je zobrazena položka dle indexu

obsaženého ve Value. Zpětná vazba kontroluje index vybrané položky a přepínacím (switch) příkazem provede akci založenou na Value. [6][10]

### 3.4.3 Vlastnosti objektu a typy funkcí

Všechny objekty mají množinu vlastností, které říkají, jak bude objekt zobrazen a jak s ním bude zacházeno. Objekt je tvořen množinou implicitních hodnot vlastností. Tyto hodnoty můžeme číst a ve většině případů i nastavit názvy vlastností. Nejsou „cAsE sEnSiTiVe“ (i tento zápis je možný), dokonce lze některé názvy vlastností i zkrátit, pokud stále vyjadřují danou vlastnost.

```
>>Figure('Color','b');
```

```
>>Figure('color','b');
```

```
>>figHandle = Figure('Position',[100 100 200 200]);
```

```
>>figHandle = Figure('pos',[100 100 200 200]);
```

Vlastnosti lze definovat dvojím způsobem:

- při vzniku objektu
- později po vytvoření objektu pomocí funkce set

#### FUNKCE SET

Pokud jsme nenastavili hodnotu vlastnosti při vzniku objektu, nebo ji chceme změnit, použijeme funkci set.

Například:

vytvoříme osy

```
>> axes;
```

Dodatečně změníme jejich barvu

```
>>set(gca,'Color','y');
```

#### FUNKCE GET

Můžeme vyvolat hodnotu vlastnosti grafického objektu.

Například:

```
PropertyValue = get(h,'PropertyName');
```

```
>>axColor = get(gca,'Color')
```

```
axColor =
```

```
1 1 0
```

## CALLBACK FUNKCE

Nad každým objektem jsou definované operace, které může uživatel využít (klik na tlačítko, výběr z nabídky, ...). Tyto operace jsou obsluhovány pomocí tzv. callback funkcí. Jinak řečeno, pokud uživatel klikne na tlačítko, aktivuje se callback funkce této události (je-li definována). Pokud nemá být GUI statický, musí vždy obsahovat alespoň jednu callback funkci. Hodnoty callback funkcí jsou uloženy jako vlastnosti objektu – lze je měnit, mazat, kopírovat...

## VYHODNOCENÍ CALLBACK FUNKCE

Zpravidla má každá eventualita vlastní callback (výběr 'Ano' má jiný callback než výběr 'Ne'). Callback funkce je vyhodnocena jako řetězec (funkcí) eval. Je vyhodnocena v základním pracovním prostoru MATLABu. [6]

### 3.4.4 Identifikátor objektů (handle)

Každý samostatný objekt má svůj identifikátor (řečí MATLABu handle). Tyto handle jsou referencí na existující objekt. Handle MATLAB vytvoří vždy, je na uživateli jeho uchování. Složitě grafy (vrstevnice) mohou mít více identifikátorů. Root má vždy handle = 0, Figure

zpravidla celé kladné číslo, ostatní objekty mají za handle kladné reálné číslo.

Handle

```
>>figHandle = Figure;
```

```
>>axHandle = axes;
```

Číslo uložené v proměnné figHandle existuje i po uzavření okna, již se ale nejedná o handle. [10]

## 4 METODY ZOBRAZENÍ MORFOLOGICKÝCH PORVRCHŮ

Pro zobrazování povrchů se využívá celá řada rozličných skenovacích metod a stále přibývají jejich další nové variace. Jejich použitelnost vychází zejména z využitého fyzikálního principu, který předurčuje pro jaké aplikace je daná metoda vhodná. Při výběru vycházíme z vlastností skenovaného materiálu, způsobu zobrazení a požadavků na přesnost a rozlišení. V neposlední řadě je také důležitá složitost realizace a ekonomická stránka. Z důvodu obsáhlosti této problematiky bude přiblížen popis mikroskopie skenující sondou, jelikož je tato metoda použita v praktické části této práce. [11]

### 4.1 Mikroskopie skenující sondou (SPM - Scanning probe microscopy)

Mikroskopie skenující sondou se vyvíjí od roku 1981, kdy Gerd Binnig a Heinrich Rohrer vynalezli skenovací tunelovací mikroskopii (za tento objev získali v roce 1986 Nobelovu cenu).

Metoda využívá přiblížení mechanické sondy k povrchu v kolmém směru. Sonda má při zkoumání buď přímý kontakt s povrchem, nebo je v jeho těsné blízkosti. Velká blízkost sondy a vzorku umožňuje snížení energie určené k měření a tím snížení energetického zatížení vzorku (zvláště v porovnání s elektronovou mikroskopií). Naopak, uvedená vzdálenost klade nároky na mechanickou stabilitu a řízení pohybu, protože může dojít k mechanickému poškození vzorku (i to lze využít, konkrétně k vytváření nanostruktur až na úrovni jednotlivých atomů).

Metoda umožňuje zobrazovat výřezy vzorku ve velikostech od stovek mikrometrů do jednotek nanometrů, pro nejmenší oblasti až se subatomárním rozlišením. Z principu metody vyplývá různé rozlišení v rovině skenování (povrchu vzorků) a ve směru k ní kolmém (označovaném jako osa  $z$ ). První z nich je dáno především velikostí sondy a měřicích rozestupů, druhé závisí na charakteru interakce a mechanické stabilitě mikroskopu. V nejlepších případech dosahují rozlišení hodnot setin až tisícín nanometru.

Metoda může sloužit také k mapování vlastností povrchů jako je teplota, hustota, vodivost, a jak již bylo zmíněno, najde také uplatnění při modifikaci povrchu.

Výhody:

- základní metody jsou poměrně jednoduché
- vysoká rozlišovací schopnost

- pořízení trojrozměrného obrazu
- mapování vlastností povrchu
- možnost modifikace povrchu

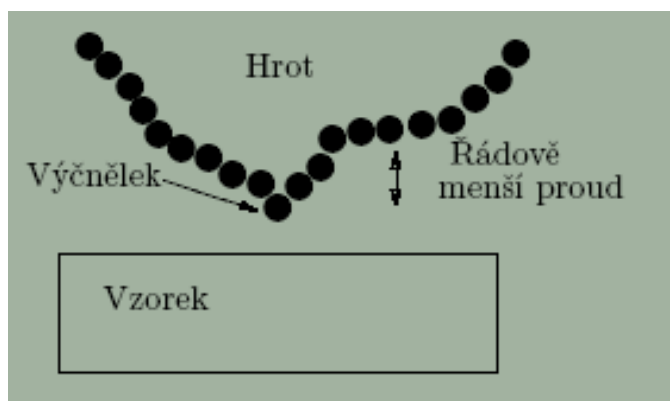
Nevýhody:

- snímání je postupné
- náchylnost k vibracím
- náchylnost na změny teploty
- možnost poškození vzorku
- k přesnému zobrazení je potřeba počítač s vhodným software

[11]

#### 4.1.1 Skenovací tunelovací mikroskopie (STM – Scanning tunneling microscopy)

První úspěšnou realizací SPM byla skenovací tunelovací mikroskopie. Je založena na monitorování proudu, který protéká mezi vodivým hrotem a vodivým vzorkem, aniž by byly v přímém mechanickém styku (viz obrázek 4). [11]



Obrázek 4 Skenovací tunelovací mikroskopie [11]

Pokud je hrot umístěn do vzdálenosti několika nm od povrchu vzorku, jsou elektrony „tunelovány“ přes tuto mezeru z hrotu k povrchu a obráceně, a to v závislosti na vzdálenosti mezi hrotem a povrchem. Tunelování se uskuteční jen v případě, kdy hrot i vzorek jsou z vodivého materiálu (případně polovodiče). V tom je zásadní rozdíl od ostatních mikroskopických metod využívajících skenovací sondu. [13]

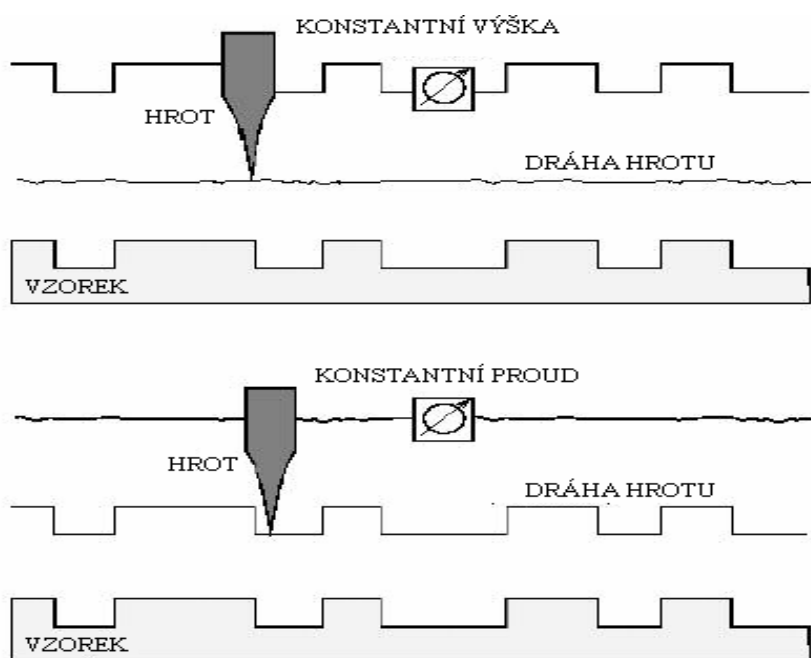
Vlastní měření probíhá tak, že nejprve se provede hrubý posuv vzorku k hrotu ve směru Z, tento může být čistě mechanický. Poté dojde k zvětšení napětí mezi hrotem a



vzorkem, aby mohl procházet proud. Nyní se jemným posuvem přiblíží vzorek ke hrotu tak, aby procházející proud nabyl měřitelných hodnot, pak se přibližování zastaví. Získání obrazu (skenování) se provádí skokovým posuvem ve dvou rozměrech (x, y) po příslušné matici měřicích bodů, zpravidla se pohybuje po řádcích a v jednom směru (zpětný pohyb je tedy prázdný). Výstupem měření je matice, jejíž indexy označují polohu bodu a příslušná hodnota je velikost měřeného signálu. Tento signál může být dvojího druhu, v závislosti na režimu měření:

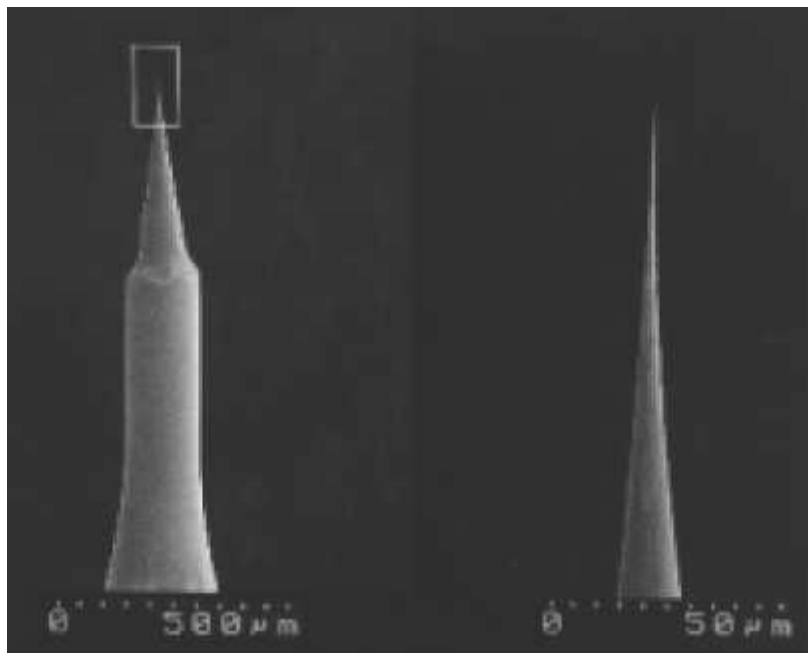
**Režim s konstantní výškou** - při němž se udržuje jednou nastavená výška a měří se velikost tunelovacího proudu. Tento režim umožňuje rychlé snímání obrazu, protože není nutno pohybovat vzorkem, ale je méně přesný, neboť při velkých vzdálenostech hrotu od povrchu se proud dostává pod dobře měřitelnou úroveň (viz obrázek 5).

**Režim s konstantním proudem** - při němž se pomocí zpětné vazby udržuje konstantní úroveň proudu. Tento režim je pomalejší, umožňuje sledovat větší změny profilu povrchu, je však závislý na převodním vztahu přiloženého napětí a změně rozměru piezo prvku. Tato závislost může být odstraněna vnějším měřičem polohy, např. laserovým. Další nevýhodou může být poškození povrchu, přejde-li hrot nad oblast s výrazně odlišnými elektrickými vlastnostmi (např. zoxidovaná místa) - aby byl udržen nastavený proud, dojde k velkému posuvu hrotu směrem dolů (viz obrázek 5). [12]



Obrázek 5 Režim s konstantní výškou a konstantním proudem [13]

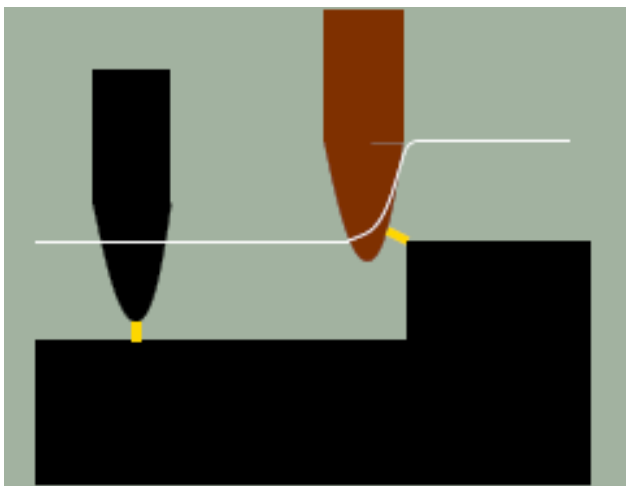
Pro dosažení vysokého rozlišení je zapotřebí mít co nejostřejší hrot (malý poloměr křivosti), nejlépe jen jeden vrcholový atom na špičce kužele, aby nedocházelo k superpozici jednotlivých interakcí. Takový hrot je však obtížné vyrobit, ale díky uvedené závislosti to ani není nutné. Postačuje hrot s makroskopickým zaoblením, je-li jeden atom vyčnívající. Přes něj pak teče téměř veškerý proud. Zde je obrázek, jak takový hrot vypadá.



Obrázek 6 Fotografie STM hrotu [11]

Rozlišení samozřejmě závisí i na velikosti skenovacího kroku, což je omezující parametr. Především v oblasti mikrometrových rozměrů. Velikost kroku nepřímo závisí na počtu měřicích bodů, který ovlivňuje dobu měření jednoho obrazu. Volba správného kroku je tedy kompromisem mezi maximálním rozlišením a délkou interakce, což má význam především při studiu dynamických jevů.

Kvůli velké blízkosti sondy je rozlišení v mikrometrových rozsazích ovlivňováno také makroskopickým tvarem hrotu, protože jednak všechny polohy sondy nenulové šířky nejsou fyzikálně možné (například není vůbec možné sledovat póry a hluboké zářezy), a jednak se mění poloha bodu nejmenší vzdálenosti od vzorku a rozchází se s polohou středu hrotu (podle kterého obraz sestavujeme).



Obrázek 7 Nejblížejší přiblížení vlivem nenulové  
šířky hrotu [11]

Žlutá barva znázorňuje oblast tunelování. Jak vyplývá z obrázku tak u hnědého hrotu dojde ke zkreslení výsledku vlivem nenulové šířky. [11][12][13][14]

## 5 POUŽITÉ PŘÍSTROJE A MĚŘICÍ TECHNIKA

V rámci realizace byla použita metoda obdobná SPM, přesněji skenování povrchu impedanční sondou za využití tunelovací mikroskopie. Měření bylo prováděné na stacionárním skenovacím hrotu pomocí digitálního multimetru Hewlett Packard 34401A a pro posuvy vzorku byly využity tři servomotory Mercury M-110 1DG, jenž zajistily posuv v osách X, Y a Z. Tyto servomotory, stejně jako digitální multimetr, byly ovládány pomocí vývojového prostředí MATLAB R2009a. [15]

### 5.1 Servomotor Mercury M-110 1DG

Servomotory jsou zařízení, které se vyznačují vysokou přesností pohybu. Na rozdíl od běžných motorů u nich lze nastavit přesnou polohu natočení osy. Ovládají se jimi například posuvy u CNC strojů nebo čtecí hlavy pevných disků (HDD). [15]

Při realizaci této práce byly využity 3 servomotory Mercury M-110 1DG od firmy Physik Instrumente, které byly k dispozici v laboratoři FAI. Tyto servomotory mají následující hodnoty udávané výrobcem:

Pohybový rozsah 5 mm

Nejmenší garantovaný krok 500 nm

Maximální rychlost 1 mm·s<sup>-1</sup>



Obrázek 8 Mercury M-110 1DG ve srovnání s 9V  
baterií [15]

Servomotory M-110 jsou ovládány přes krokové ovladače Mercury C-862 Mercury Networkable Single-Axis DC-Motor Controller. Tyto krokové ovladače slouží k flexibilnímu řízení pohybů servomotorů. Každý ovladač může ovládat pouze jeden motor. Tyto ovladače se dají vzájemně propojovat, čímž je umožněno adresně ovládat až šestnáct servomotorů. Pohyb posuvů je realizován odesíláním a přijímáním textových řetězců obsahujících identifikační a řídicí znaky přes port RS-232 mezi počítačem a krokovými ovladači C-862. [15]

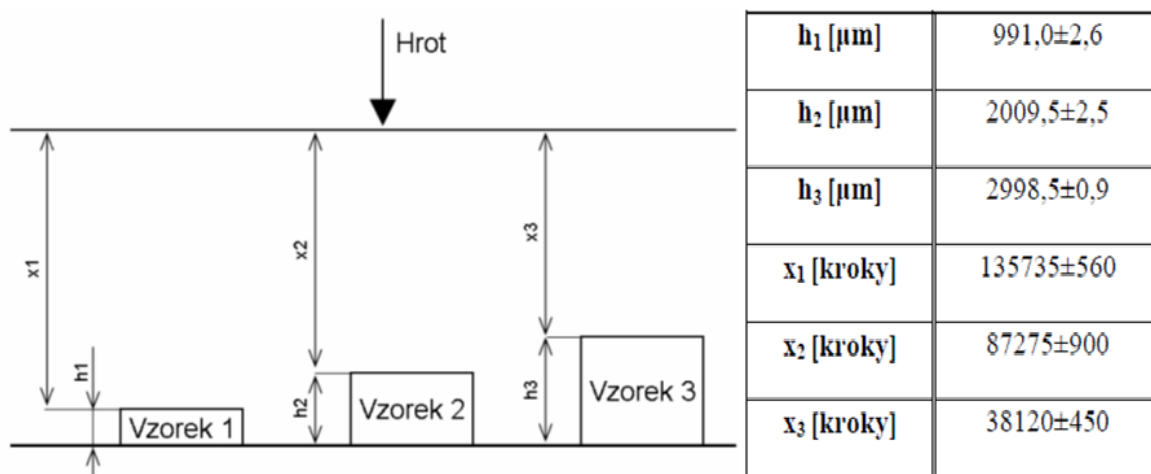


Obrázek 9 Mercury C-862 ve srovnání s mincemi 1\$ a 1€ [19]

### 5.1.1 Velikost jednoho kroku posuvu servomotoru

Tímto tématem se již zabýval student Josef Kudělka ve své diplomové práci, proto jej budu pouze citovat. Na základě jeho měření budeme považovat délku jednoho mikrometru jako 49 kroků (průměr z jeho tří měření, viz obrázek 10).

*„Při komunikaci se servomotory nejsou zadávány hodnoty posuvů, které se mají vykonat, v metrických jednotkách, ale v tzv. krocích. Krok je relativní míra, jenž se mění dle nastavení měřicí soustavy. K odhadnutí velikosti jednoho kroku bylo rozhodnuto při stejném nastavení měřicí soustavy provést sérii měření na vzorcích, u nichž budou známe jejich přesné rozměry. Na základě předchozích měření byla vypočítána velikost jednoho kroku. Z důvodu kontroly byly použity výsledky měření všech tří vzorků.“*



Obrázek 10 Přehled provedených měření a přehled naměřených hodnot [15]

*Odhadnutí velikosti jednoho kroku na základě měření vzorku 1 a 2 :*

*Rozdíl mezi  $h_2$  a  $h_1$  je  $(1019 \pm 4)$   $\mu\text{m}$ . Rozdíl mezi  $x_1$  a  $x_2$  je  $(48460 \pm 1100)$  kroků. Z toho vyplývá:  $1 \mu\text{m} \approx 48$  kroků.*

*Obdobným způsobem byla odhadnuta velikost jednoho kroku na základě měření vzorků 2 a 3 a následně 1 a 3.*

*Odhadnutí velikosti jednoho kroku na základě měření vzorků 2 a 3:*

*$1 \mu\text{m} \approx 50$  kroků*

*Odhadnutí velikosti jednoho kroku na základě měření vzorků 1 a 3:*

*$1 \mu\text{m} \approx 49$  kroků*

*Z uvedených výsledků vyplývá, že měřením na všech vzorcích bylo dosaženo navzájem se potvrzujících výsledků.“ [15]*

## 5.2 Digitální multimetr - Hewlett Packard 34401A

Pro automatizaci a zpřesnění celého procesu měření a softwarového algoritmu byl využit digitální multimetr Hewlett Packard 34401A. Tento digitální multimetr byl připojen k PC přes rozhraní GPIB (komunikační protokol Visa) a ovládán opět přes vývojové prostředí MATLAB. [15]

Digitální multimetr 34401A od firmy Hewlett Packard má maximální rozlišení  $6\frac{1}{2}$  digitu. Relativní nejistota měření je pro střídavé hodnoty 0,06% a pro hodnoty stejnosměrné 0,0015%. Přístroj umožňuje měření až do 1 kV. Šířka měřicího pásma je 3

Hz až 300 kHz. Rychlost měření výrobce uvádí až 1000 měření·s<sup>-1</sup> v závislosti na počtu použitých digitů při měření. [16][18]



Obrázek 11 Přední panel digitálního multimetru Hewlett Packard 34401A [17]

### 5.3 Možnosti zvýšení přesnosti měření - využití P-611.3S NanoCube

Pro získání většího rozlišení je možné k již sestavené soustavě přidat P-611.3S NanoCube, rovněž od firmy Physik Instrumente. Nanocube P-611.3S umožňuje pohyb ve třech osách s rozsahem 120x120x120  $\mu\text{m}$ . Rozlišovací schopnost je 0,2 nm. K ovládání je potřeba krokový ovladač. Škola vlastní Position Servo Controller E-664.



Obrázek 12 P-611.3S NanoCube [14]



Obrázek 13 P-611.3 NanoCube v kombinaci se soustavou M-111 1DG [14]



## **II. PRAKTICKÁ ČÁST**

## 6 SEZNÁMENÍ S PROGRAMOVACÍM PROSTŘEDÍM MATLAB

Pro práci bude důležité znát všechny uicontrol objekty a umět s nimi pracovat. Ve zkratce budou představeny důležité prvky a jejich účel (okna, objekty, funkce). Dále je nutné zmínit tagy, což jsou vlastnosti objektu, které jasně identifikují daný objekt a jsou využívány pro propojení s m-soubory.

### Simulink

Simulink je nástroj, který využívá MATLAB a jeho funkce k simulaci dynamických systémů. Simulink má jiné uživatelské rozhraní než MATLAB. Zatímco u MATLABu je stále nejdůležitější příkazový řádek, ovládání Simulinku je jednodušší a intuitivnější, ale pokročilejší funkce nelze provádět bez znalosti jazyka MATLAB. Interaktivní způsob tvorby a simulace modelů se spouští z příkazové řádky systému MATLAB příkazem simulink. Po spuštění je vytvořeno okno pro tvorbu nového modelu a okno obsahující základní nabídku otvírání knihoven zdrojů signálů, základních spojitých, diskrétních a nelineárních bloků a bloků pro zobrazování a ukládání signálů. Pod touto interaktivní obálkou se skrývá systém velmi podobný grafickému subsystému s obdobnými funkcemi simget a simset. Další vrstva funkcí umožňuje již komfortnější neinteraktivní tvorbu modelů systémů. Pro obvyklého uživatele však není nutné o implementaci a programování modelů přemýšlet.

### Profiler

Profiler je nástroj pro optimalizaci kódu. Můžeme zjistit, která část programu je časově nejnáročnější, abychom mohli kód dále optimalizovat.

### M-soubory

Dávkové soubory, jsou to samostatné soubory s uloženým kódem, které můžeme volat a poté vykonat.

### GUIDE

Guide je nástroj pro tvorbu grafického rozhraní, je alternativou profileru, je to soubor s koncovkou \*.fig a určuje grafickou stránku výsledného grafického uživatelského rozhraní.

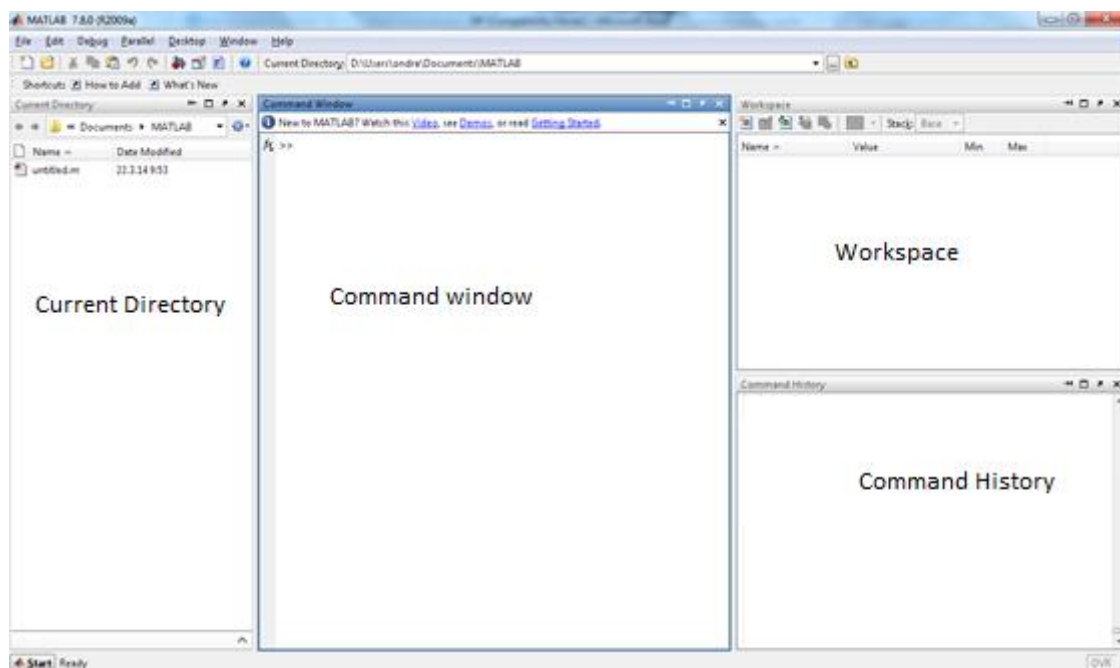
- umožňuje vytvářet a editovat uživatelské rozhraní pomocí základních komponent (checkbox, sliders, tables apod.)

- všechny komponenty, které jsou vytvořeny v tomto prostředí, lze měnit za běhu aplikace
- vzhled vytvořené GUI aplikace je ukládán do souboru s příponou \*.fig a jeho zdrojový kód s příponou \*.m

Spuštění průvodce pro tvorbu GUI aplikací je možné více způsoby. Jedním z nich je využít základní menu File/New/GUI nebo zápisem a potvrzením příkazu GUIDE v hlavním prostředí MATLABu.

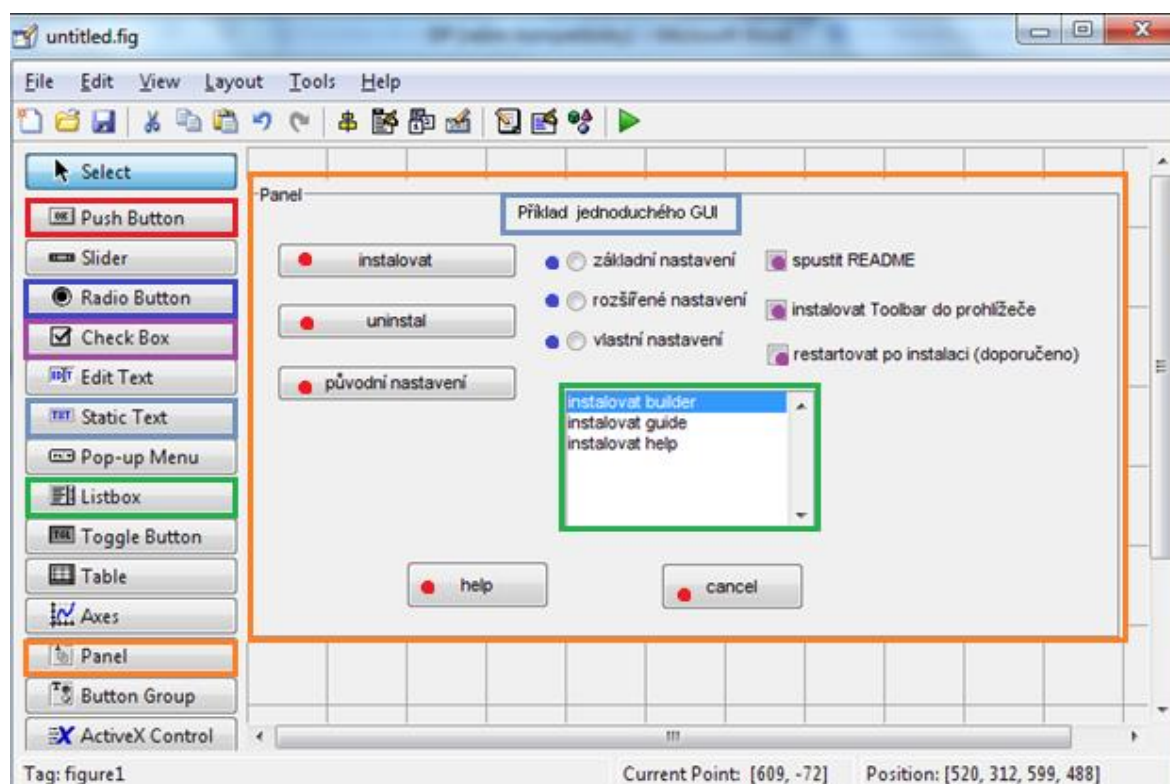
Program MATLAB se skládá z několika pevných a několika volitelných oken. Ty si můžeme zmenšit, posunout, nebo úplně minimalizovat a taky zavřít. Nejdůležitější z nich je Command Window, který slouží jako příkazový řádek (CLC), dále slouží k výpisu kódu, odpovědí programu, výpočtů a podobně. Okno Current Directory slouží jako průzkumník pro pohyb mezi složkami a soubory. Jejich následné spouštění, úpravy a práce s nimi. Okno Workspace nám zobrazuje proměnné, jejich název a formát. Programátor má přehledně zobrazeny jednotlivé proměnné, s kterými pracuje a může je velmi snadno editovat, mazat, tvořit nebo s nimi jinak pracovat. Dalším oknem je Command History. Slouží pro ukládání provedených kódů. Jednoduše výběrem jakéhokoli kódu z Command History a poklepnutím na něj se kód provede. Můžeme pomocí „ctrl“ a kliknutí myši vybrat více příkazů a dále s nimi pracovat. Command History nabízí zajímavé funkce, například zvolené příkazy uložit do nového m-souboru. Grafické znázornění jednotlivých oken je na obrázku č. 14.

Další volitelné okno je Help. Jedná se o klasickou nápovědu k programu. Buď otevřeme okno help a projdeme si veškerou nápovědu, popis i s příklady, nebo můžeme využít příkazového řádku, kdy zapsáním „help“ mezera „příkaz“ se nám vypíše popis přímo do příkazového okna. Editor slouží k úpravě m-souborů. Jsou zde označeny čísla řádků, kód je barevně rozdělen pro snadnější orientaci. Zelenou barvu mají komentáře, růžovou vypisovaný text, modrou funkce a černou zbytek kódu.



Obrázek 14 Program MATLAB, základní okna a jejich popisky

Další součástí je pak grafické uživatelské rozhraní tvořené skrze funkci GUIDE (voláme funkci `>>GUIDE`). Popis a grafické znázornění uicontrol prvků je na obrázku 15.



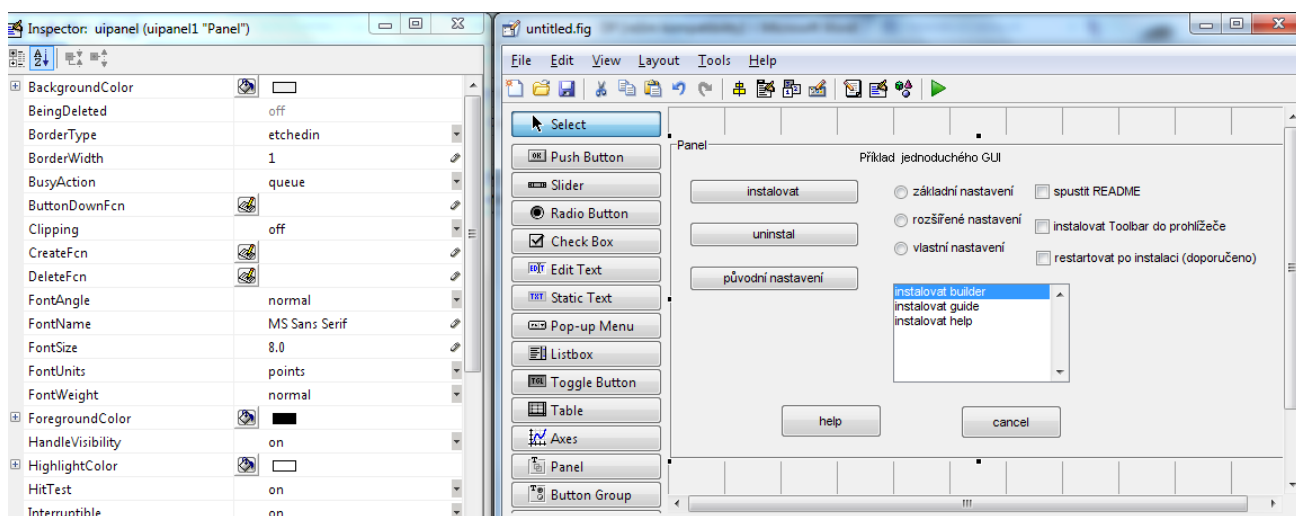
Obrázek 15 Grafické znázornění jednotlivých uicontrol prvků

## 6.1 Návrh a tvorba jednoduchého GUI

Jednoduše si můžete vytvořit obecné GUI. Skrze příkaz GUIDE nebo ikonu na hlavním panelu můžeme spustit GUIDE. Pomocí objektově orientovaného programování můžeme vytvořit GUI. Jednoduše přetáhnete pořadované prvky (objekty) z panelu na levé straně na požadované místo, můžete využít zarovnávání a dalších funkcí, aby GUI dostalo uživatelsky příjemný vzhled. Takovéto GUI však ještě není schopno plnit příkazy. Je k tomu potřeba naprogramovat jednotlivé tlačítka, check boxy, slidery, prostě uicontrol prvky. Spojení dávkového souboru (m-souboru) s Figure je velmi důležité. Dosáhneme toho zapsáním kódu do m-souboru a propojením s GUI skrze tagy jednotlivých objektů zajistíme spojení s naprogramovanými funkcemi. Taky najdeme v položce properties (vlastnosti) daného prvku. Poté otevřeme dávkový soubor (m-soubor) a najdeme požadovanou funkci, do které zapíšeme kód, který se má provést při použití objektu.

## 6.2 GUIDE

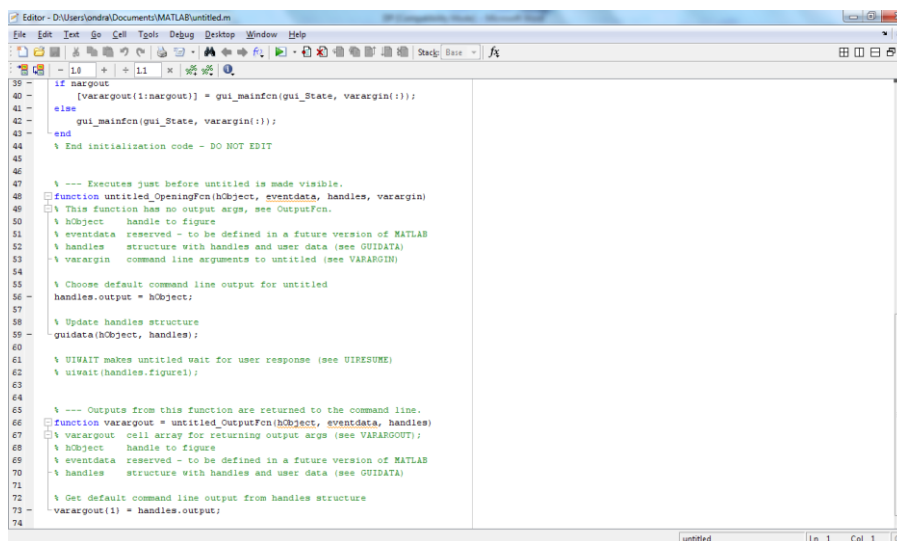
Nástroj GUIDE po tvorbu grafického uživatelského rozhraní uloží výsledný grafický výstup jako prázdné okno s názvem Figure. Když soubor uložíme a dáme mu název, vytvoří se soubor s koncovkou (\*.fig). Po uložení Figure se nám vytvoří m-soubor s již zadanými uicontrol objekty se stejným názvem a koncovkou pro m-soubor (\*.m). Poté je zapotřebí upravit kód jednotlivých funkcí. Na obrázku 16 vlevo je znázorněno okno vlastností uicontrol prvku (properties), na pravé straně je zobrazen GUIDE.



Obrázek 16 GUIDE – nástroj pro tvorbu uživatelského rozhraní

### 6.3 Dávkové soubory (m-soubory, m-files)

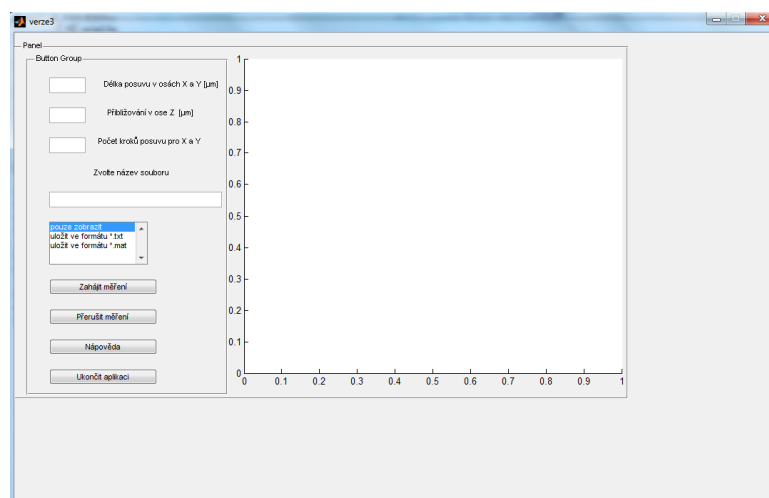
Pro dávkové soubory musíme vytvořit nový m-soubor. Toho dosáhneme přes nabídku File -> New -> Open blank m-file. Poté se nám otevře nové okno (viz obrázek 17).



Obrázek 17 Dávkový soubor (m-soubor)

### 6.4 Figure

Figure, výsledný produkt GUIDE je pak samostatné okno, které při spuštění (funkce RUN) funguje jako grafické uživatelské rozhraní schopné reagovat na vstupní podněty (skrže vstupní zařízení – klávesnici, myš, porty, rozhraní a podobně), zpracovat je, vytvářet výstupy (data, zobrazovat dané funkce, počítat funkce). Figure je zobrazeno na obrázku 18.



Obrázek 18 Zobrazení ve figure

## 7 NÁVRH A REALIZACE GUI PRO OVLÁDÁNÍ MIKROPOSUVŮ

Je potřeba prvně zmínit obecné informace o skenovací mikroskopii odporovou sondou, požadavky na ni, jaké předměty jsou vhodné k měření, popis použitých přístrojů a jejich funkce, omezení a možnosti, abychom mohli blíže pochopit jednotlivé části programu, jednotlivé proměnné a zadávané hodnoty a funkce.

### 7.1 Obecné informace o měření a použitých zařízeních

Naše měřicí soustava se skládá ze sériové linky, měřicího přístroje (digitálního multimetru), impedanční sondy, kabeláže a servomotoru pro mikroposuvy.

Impedanční sonda je upevněna ke stojanu. K ní je připojena kabeláž připojena na digitální multimetr. Ten je připojen přes sériové rozhraní k počítači. K počítači jsou dále připojeny servomotory, které se pohybují ve třech osách pod měřicí sondou. Teď, když máme základní představu jak vypadá náš měřicí přístroj pro skenování, můžeme definovat jednotlivé vstupní parametry pro měření a dle toho navrhnout výsledný vzhled GUI.

#### 7.1.1 Využitelnost měření

Jelikož zařízení pracuje na principu měření proudu, je možné proměřovat pouze vodivé předměty nebo předměty pokovené.

#### 7.1.2 Zadávané hodnoty, délka kroku

Hodnoty pro délky posuvu mezi dvěma zadanými body v ose X, Y, Z jsou zadávány v přepočtu na mikrometry. 49 otáček hřídele servomotoru je přibližně 1  $\mu\text{m}$ . Přičemž nejmenší garantovaný krok je 500 nm, nejmenší garantovaný krok je  $\approx 25$  otáček hřídele servomotoru. Program pracuje pouze s celými čísly. Pro jednoduchost budeme brát, že 1  $\mu\text{m}$  je 50 otáček hřídele servomotoru. Vkládané hodnoty délky kroku mezi dvěma body jsou zadávány v mikrometrech kvůli uživatelsky srozumitelnějšímu sdělení. Pro plné využití garantovaného kroku by byla nejmenší hodnota kroku 0,5. [15]

## 7.2 Návrh GUI

Při návrhu GUI je potřeba zohlednit požadavky na grafické uživatelské rozhraní (kapitola 3.1) a návrh bude proveden dle kapitoly „Proces návrhu designu“ (kapitola 3.2). Je tedy potřeba prvně definovat požadavky na GUI, poté ho navrhnout, otestovat design, propojit s kódem a kód následně otestovat, proti vzniku chyb.

### 7.2.1 Definovat požadavky na GUI

Grafické uživatelské rozhraní musí splňovat základní obecné požadavky jako je provázanost, komplexnost (ošetření všech eventualit), jednoduchost, pohodlnost, přívětivost, přehlednost a další (dle kapitoly 3.1). Další požadavky jsou pro získání vstupních parametrů pro samotný měřicí program.

Sonda pro skenovací mikroskopii vykonává pohyb ve třech osách. V ose X, v ose Y a v ose Z. Tyto hodnoty je potřeba definovat. Dále je potřeba hodnota „krok“, která určí délku mezi jednotlivými body v ose X a Y. Je požadována inicializace přístrojů před měřením mikroposuvy, inicializace portů a sériového rozhraní, je potřeba provést restart pozice měřicí sondy. Další vyžadovaná nabídka je volba výstupního formátu dat. Tlačítko „Ukončit měření“ které ukončí měření v jakémkoli stádiu měření a provede bezpečný návrat měřicí sondy do základní pozice bez poškození sondy. Tlačítko „Zahájit měření“ pro inicializaci příkazů měření a konec programu. Je potřeba v GUI zimplementovat tyto požadavky.

- „Inicializace přístrojů“
- „Reset pozice“ pro návrat skeneru do původní pozice
- Pole pro získávání vstupních dat (počet bodů v ose X, Y a délka posuvu pro osy X, Y a Z, název souboru a typ)
- Nabídka „výstupní formát dat“
- Tlačítko „Zahájit měření“, které provede nastavené příkazy
- Tlačítko „Ukončit měření“, které ukončí program v kterékoli fázi měření
- Tlačítko „Nápověda“
- Ukončit program

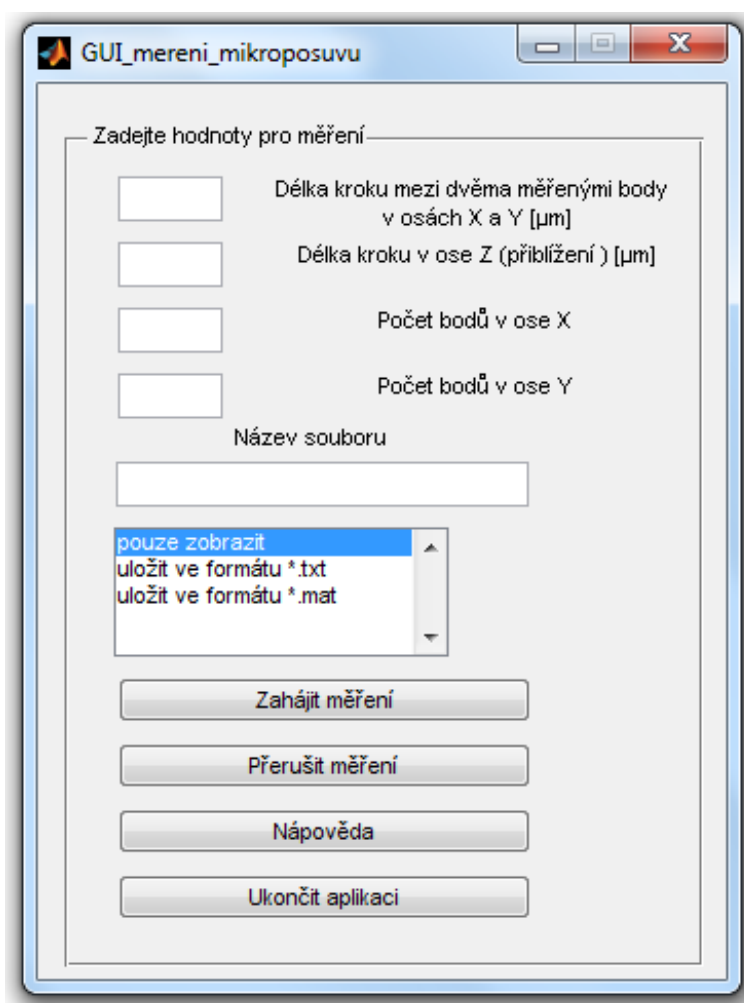


### 7.2.2 Návrh GUI – vhodné uicontrol objekty

Využité uicontrol objekty:

- PUSH BUTTON – tlačítka Zahájit měření, Ukončit měření, Nápověda, Ukončit program
- STATIC TEXT – Popisy pro okna vstupních parametrů
- EDIT TEXT – pro vstupní hodnoty funkcí mikroposuvu (pro osy X, Y a krok posuvu v ose X-Y a Z a počet měřených bodů v ose X a Y)
- LISTBOX – pro výběr výstupního formátu dat
- PANEL – sjednocení jednotlivých prvků do „obalu“
- BUTTON GROUP – seskupení tlačítek

### 7.3 Výsledný vzhled GUI



Obrázek 19 Výsledný vzhled GUI

## 7.4 Jednotlivé části programu

Program se skládá z několika částí. Je to měřicí část programu, zobrazení průběhu měření, inicializační část programu, reset pozice a smyčka kontrolující přerušení programu.

### 7.4.1 Měřicí část programu

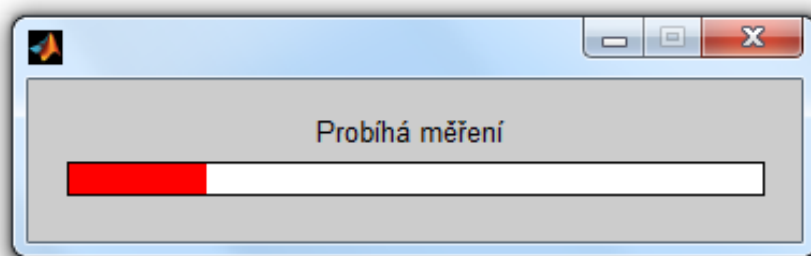
Měřicí program pracuje ve smyčce, kdy se servomotor s měřeným vzorkem přibližuje v ose Z do té doby, než se změní měřená hodnota odporu mezi sondou a měřeným vzorkem z nekonečna na hodnotu v zadaném rozsahu. Ve výchozí pozici je odpor nekonečně veliký (měřicí sonda se nedotýká měřeného vzorku, ležící na vodivé podložce). Jedna ze svorek měřících odpor je upevněna k měřicí sondě, druhá je upevněna k vodivé podložce. Jestliže se měřicí sonda dotkne měřeného objektu (předmět musí být vodivý), hodnota odporu se změní a program zapíše hodnotu přiblížení při změně hodnoty odporu. Poté se servomotor posune o jeden bod dále v ose X a měření se opakuje. Jakmile jsou proměřeny všechny body v ose X, servomotor se posune v ose Y o jeden bod a opět se opakuje proměření všech X bodů. Takto se pokračuje, dokud nejsou proměřeny všechny body.

### 7.4.2 Inicializační část programu

Skládá se z čtyř částí. Jedná se o inicializaci sériové linky, inicializaci servomotoru, nastavení výchozí pozice a inicializaci měřicího přístroje (digitálního multimetru). Inicializace probíhá odesíláním řídicích dat po sériové lince do jednotlivých přístrojů.

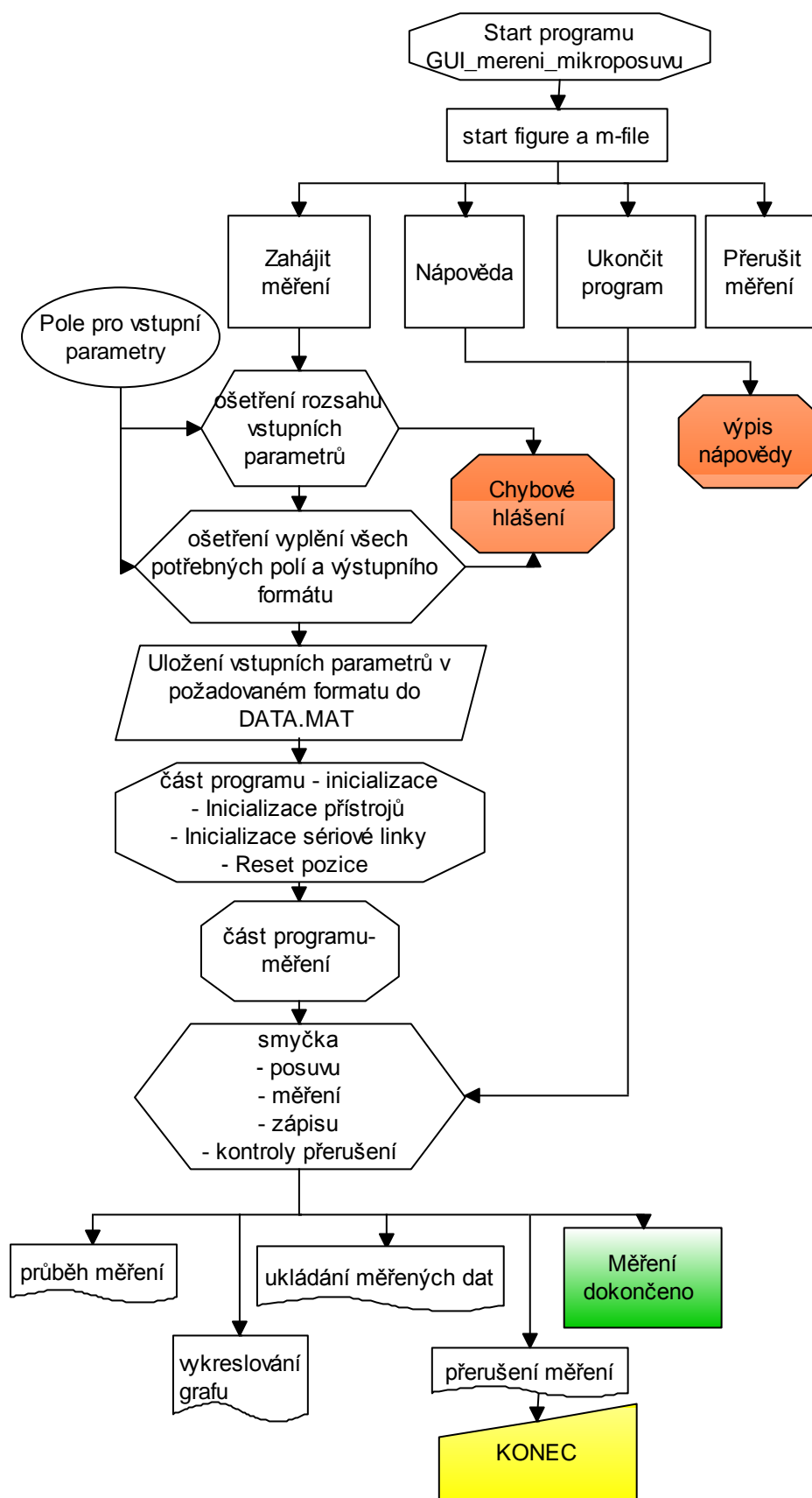
### 7.4.3 Průběh měření

Tato část programu je realizována funkcí waitbar, která skrze vstupní proměnnou vyjadřuje průběh měření. Tato proměnná může nabývat hodnot 0-1. Proto vyjádříme pouze poměr mezi aktuální měřenou pozicí a celkovým počtem měřených hodnot. Tento poměr nám vstupuje jako vstupní proměnná funkce waitbar, která vyjadřuje, v jaké části se program nachází.



Obrázek 20 Funkce waitbar pro znázornění průběhu měření

## 7.5 Schéma algoritmu programu



Obrázek 21 Schéma algoritmu programu

## 7.6 Test designu

Při tvorbě GUI byl kladen důraz na základní principy pro tvorbu GUI. A to zejména jednoduchost, přehlednost, intuitivnost. Program je vzhledově jednoduchý, srozumitelný, přehledný a zároveň obsahuje všechny nezbytné důležité funkce. Možnosti zobrazení a úpravy jsou možné v okně Figure. Uživatel tedy může upravit vzhled, popisky os, legendy, zobrazení grafu a úhle pohledu dle svých požadavků.

Při postupném testování programu se ukázaly být některá tlačítka nadbytečná. Například inicializace přístrojů, inicializace sériové linky, reset pozice při měření. Tyto všechny podprogramy stačí spustit jednou na začátku programu. Poté teprve následuje provedení samotného měřicího programu na základě požadavku stisknutí tlačítka „Zahájit měření“

## 7.7 Zápis kódu a propojení s GUI

Propojení GUI a kódu je realizováno skrze tagy na jednotlivé uicontrol prvky v GUIDE. V m-souboru je zapsán příkaz, který se má provést. Po aktivaci daného prvku (například tlačítka) se provede kód definovaný ve funkci daného uicontrol prvku.

## 7.8 Test kódu a ošetření proti chybám

Tato část je při programování nejsložitější a provází programátora od začátku programu. Je třeba veškerý přidaný kód otestovat, jestli je správně zapsán, jestli po výpočtu jsou data ve správném formátu atd. Zde je důležité eliminovat veškeré chyby, jak ze strany uživatele, tak ze strany programátora nebo programu. Veškeré chyby odhalí sám program a přesměruje nás na řádek, na kterém se vyskytuje chyba. To velmi ulehčí práci s opravou kódu.

Proti chybám ze strany uživatele byly vytvořeny chybové hlášení při nesplnění daných podmínek.

Výčet jednotlivých ošetření programu proti chybě:

- Vyplnění všech potřebných polí
- Nesprávně zvolené hodnoty – mimo rozsah, text namísto čísla
- Inicializace portů – při obsazení portů vypíše chybovou hlášku

- Inicializace multimetru – v případě, že multimetru neodpovídá je vypsáno chybové hlášení
- Inicializace servomotoru – v případě, že multimetru neodpovídá je vypsáno chybové hlášení
- Ošetření proti přepsání souboru (zadání názvu již existujícího souboru)
- Ukončení aplikace – znovu dotázání, jestli si přejete aplikaci ukončit
- Přerušování měření – znovu dotázání, zda si přejete přerušit měření

Nevyplnění všech potřebných polí (název, počet bodů v ose x, počet bodů v ose y, přibližování v ose Z). Jestliže nejsou vyplněna všechna požadovaná pole, vyskočí varovná hláška „Varování, nejsou vyplněna všechna požadovaná pole“.

Uživatel zadá hodnotu mimo rozsah, nebo v jiném než číselném formátu. Toto je ošetřeno pomocí kontroly, zda hodnota je číslem a jestli je v požadovaném rozsahu. Jestliže tato podmínka není splněna, vyskočí chybové hlášení.

Inicializace portů, měřicího přístroje (digitálního multimetru), servomotoru. Toto je ošetřeno skrze zachycení dané chyby a výpis do okna varování (funkce warndlg).

Ošetření proti přepsání souboru je realizováno skrze dotaz na existenci souboru se stejným jménem.

Ukončení aplikace a přerušování měření je ošetřeno tak, že se opětovně zeptá, jestli si přejeme provést daný krok.

Proti chybám ze strany programu nebo programátora, zde se jedná o to, jestli programátor zná dostatečně program. Stává se, že program automaticky nastaví typ proměnné a při dalších operacích s nimi je může změnit na jiný typ proměnné. Tak se může stát, že hodnotu, kterou v programu několikrát měníte (např její typ z char na number nebo string) se stane pro program nečitelná. Dynamické typování proměnných může být často výhodou, ale když se vyskytne problém, těžko se tato chyba hledá, protože program odkazuje na jinou část kódu, než kde je definovaná proměnná.

## 8 VOLBA VHODNÉHO ZPŮSOBU UKLÁDÁNÍ DAT A JEJICH VIZUALIZACE

Program MATLAB disponuje možností vizualizace dat. Je proto možné měřené data zobrazit přímo v programu. V případě, že uživatel nebude spokojen s vizualizací v programu MATLAB, je zde možnost zvolit výstupní formát textového souboru (\*.txt), nebo souboru data MATLAB (\*.mat) a další vizualizace v jiném grafickém programu.

### 8.1 Způsob ukládání dat

V GUI byly vytvořeny 2 možnosti ukládání dat a 1 možnost uložit do souboru, který je při nezvolení formátu vybrán. Mimo GUI je možnost zkopírovat vypisovaná data z okna Command Window programu MATLAB. Data jsou ukládána do matice. Ukládání dat probíhá dynamicky při měření. Ve smyčce se mění hodnota měřené pozice a podle aktuální pozice je zapsána hodnota na místo, kde byla měřena. Pro představu to může vypadat takto:

Matice(m,n)=z

m – je index řádku

n – je index sloupce

z – je naměřená hodnota

Prvky matice jsou indexovány následovně:

Tabulka 1 Indexace při zápisu do matice

M(m,n)				M(m)			
1,1	1,2	1,3	1,4	1	11	21	31
2,1	2,2	2,3	2,4	2	12	22	32
3,1	3,2	3,3	3,4	3	13	23	33
4,1	4,2	4,3	4,4	4	14	24	34
5,1	5,2	5,3	5,4	5	15	25	35
6,1	6,2	6,3	6,4	6	16	26	36
7,1	7,2	7,3	7,4	7	17	27	37
8,1	8,2	8,3	8,4	8	18	28	38
9,1	9,2	9,3	9,4	9	19	29	39
10,1	10,2	10,3	10,4	10	20	30	40

	1	2	3	4	5	6	7	8	9	10
1	15470	17430	17780	17920	18200	18830	18900	18340	18550	19000
2	15470	18550	17640	8890	18200	18060	18550	18690	19530	19000
3	15890	17570	17500	17710	18060	18270	1470	18760	19040	19000
4	15610	17710	17430	17710	17920	18410	5110	19040	18340	18000
5	16450	16940	17780	17500	18410	18130	18620	18760	19110	18000
6	15330	16730	17640	13300	18620	17640	18970	18760	18550	17000
7	16100	17080	17220	17150	17430	17640	17850	18130	18550	18000
8	15750	17150	17150	18060	17430	18200	17710	18060	18690	18000
9	16100	16870	17570	12110	17850	17780	17710	17990	18060	18000
10	16030	17290	17430	980	18200	18060	17640	18060	18130	18000
11	17150	17080	17710	16800	18200	18200	17780	17710	18550	18000
12	15470	17850	17290	17710	17710	18200	18340	18060	17990	18000
13	15540	17920	18200	17290	17780	17850	18270	18340	18130	18000
14	15750	17080	17010	17710	18200	17850	6790	17780	17850	18000
15	16590	17150	17640	3010	17850	17710	17640	17990	18340	18000
16	15540	16660	17430	14280	17360	18130	17850	17920	17920	18000

Obrázek 22 Struktury ukládaných hodnot do matice v programu MATLAB

Uživatel může zvolit z tří možností zápisu dat.

- Volba pouze zobrazit

V tomto případě se automaticky vytvoří soubor „pouzezobrazit.mat“, který dočasně uloží hodnoty potřebné pro vykreslení grafu. Při příštím měření a volbě pouze zobrazit je tento soubor smazán a znovu vytvořen nový.

- Volba uložit ve formátu \*.txt

Pro tuto volbu jsou data uložena jako matice hodnot v textovém souboru.

- Volba uložit ve formátu \*.mat

Pro volbu uložit ve formátu \*.mat je nastaveno ukládání hodnot do souboru data MATLAB. Data jsou rovněž ukládána jako matice hodnot. Tato volba je nejlepší pro další práci s daty v programu Matlab. Umožňuje strukturované a přehledné zobrazení přímo v okně workspace.

## 8.2 Vizualizace dat

Vizualizace dat v GUI programu je řešena volbou zakreslování do grafu dynamicky (dynamicky, okamžitě po měření) nebo data pouze uložit a poté je teprve zobrazit vybraným programem. Na to slouží prvek checkbox.



Vizualizace dat v programu Matlab probíhá v okně Figure. Pro vykreslení 3D povrchu jsme využili funkci `surf` (surface). Tato funkce nám otevře nové okno Figure, v kterém se nám vykresluje graf naměřených hodnot. Můžeme dále měnit parametry zobrazení, což je velkou výhodou. Můžeme změnit rozsah hodnot, pojmenování os, zobrazit legendu, zvolit jiné barevné schéma pro vykreslování, s grafem rotovat, zkrátka nastavit si ho dle svých potřeb.

Dále je možnost využití jiného programu pro zobrazení dat (například Wolfram Mathematica). Je to velmi kvalitní program a nabízí širokou paletu možností zobrazení a nastavení.

### 8.3 Příklad vizualizace již dříve naměřených dat

Měřená data jsou zobrazována automaticky vykreslováním do grafu. Jestliže však chceme vizualizovat již dříve měřená data, musíme v příkazovém řádku MATLABu použít následující postup.

Naměřené data jsou uložena v souboru, jehož název jsme definovali v GUI (pro příklad ho nazvěme `mince40.mat`). V tomto souboru je uložena matice hodnot pod názvem „Matice“. Proto při vizualizaci uložených souborů je nutné použít příkaz:

```
>>load mince40.mat
```

```
>>surf(Matice)
```

To nám zajistí vizualizaci naměřených dat.

## 9 OVĚŘENÍ FUNKČNOSTI NAVRŽENÉHO PROGRAMU

Měření bylo nastaveno a spuštěno skrze grafické uživatelské rozhraní. Po zadání vstupních parametrů (délku kroku posuvu v ose X a Y, přibližování v ose Z, počet bodů v ose X, počet bodů v ose Y, název a datový typ) byla spuštěna inicializační část programu, která inicializovala porty, digitální multimetr a servomotory. Poté proběhl návrat servomotorů do výchozí pozice. V této části začíná měření vybraného vzorku.

Program byl testován sérií měření. Funkčnost finálního programu byla bezproblémová. Problémy, které vznikaly, byly průběžně opravovány. Problémy vznikaly zejména při měření, mimo programovou část. I tyto problémy byly z velké části eliminovány.

### 9.1 Vzniklé problémy a řešení – programová část

#### Nedostatečná výpočetní kapacita

Při měření mince 100x100 bodů vznikl problém ve výpočetní kapacitě počítače, na kterém se provádí měření. Počítač, na kterém bylo prováděno měření je dosti starý a při nastavení dynamického vykreslování do grafu se čas měření enormně prodloužil. Odhadovaný čas měření bez vykreslování byl 4 dny 12 hodin (průměrně 40 s na naměření jedné hodnoty, měřených hodnot 10 000), avšak při dynamickém vykreslování byl čas dvakrát delší. Proto jsem do programu přidal možnost graf nevykreslovat a pouze uložit.

#### Opětovná inicializace – obsazené porty

Při opětovné inicializaci portů vznikl problém, že porty byly již přiděleny při předchozím volání funkce. Pro uvolnění portů musel být program resetován.

#### Vstupní hodnoty pro měření mimo požadovaný rozsah

Tento problém musel být ošetřen přímo v programu kontrolou rozsahu. Měřená plocha může být maximálně 5x5 milimetrů.

### 9.2 Vzniklé problémy a řešení – měřicí část

#### 9.2.1 Chyby měření

- Měření v okolí vodivých hran
- Špatně zvolený krok (moc veliký, vzrůstá nepřesnost)

- Nedokonale očištěný povrch vodivých součástí (měřicí podložky, měřené vzorky, měřicí sondy)
- Špatný kontakt svorek na sondě nebo na měřicí podložce servomotoru
- Vibrace, otevírání dveří (změna tlaku)

Chyby měření byly nejčastěji způsobeny vibracemi. Měřicí přístroj je umístěn na stole. Při kontaktu se stolem se tak přenášejí vibrace, které způsobují nepřesnosti. Sousedící stůl se také dotýká stolu, na kterém probíhá měření. Na sousedním stole je umístěn počítač, monitor a další vybavení, které způsobují vibrace (ventilátory, servomotory a podobně). Při otevírání dveří vzniká změna tlaku, která může mít na měření také vliv, při zavření dveří vznikají vibrace také. Další podstatné zkreslení vzniká stanovením nesprávně zvoleného měřicího kroku.

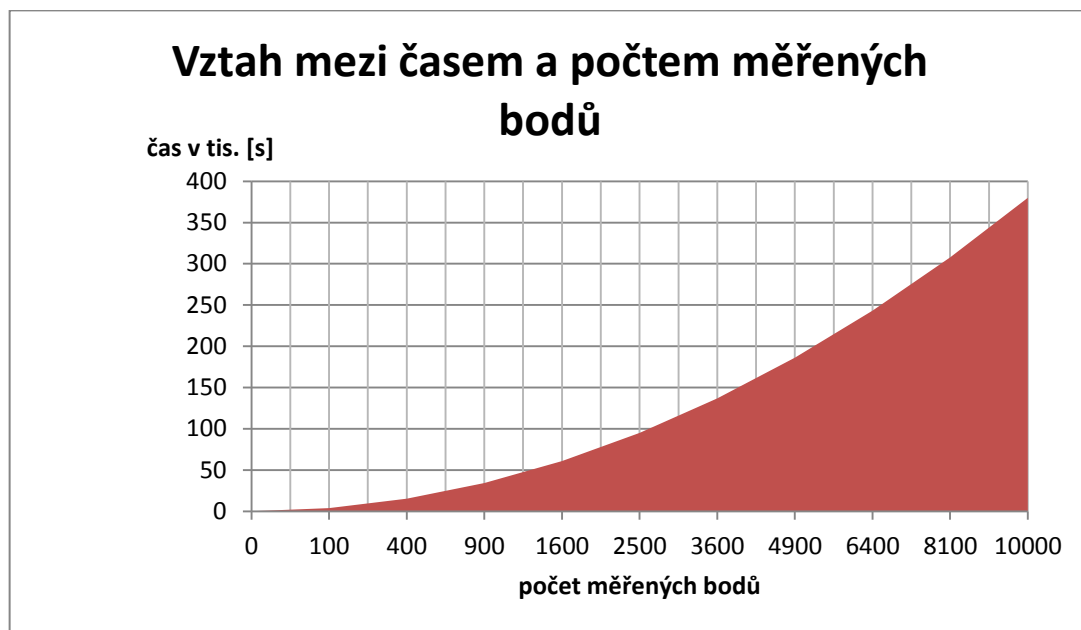
Pro přesné měření je důležité zvolit dostatečně přesný krok. Platí, že čím více bodů na měřené ploše, tím lépe a taky čím menší přibližovací krok, tím je měření přesnější. Povrch měřeného vzorku, měřicí sondy a měřicí podložky servomotoru musí být důkladně mechanicky a chemicky očištěny (alkoholem vůči oxidaci a jiným typům nečistot). Je potřeba zkontrolovat měřicí svorky, jestli jsou správně upevněny. Pro přesné měření by bylo potřeba zamezit přenosu vibrací, což se v našem případě nepodařilo. Místnost slouží jako pracovna pro studenty doktorandského studia. V okolních místnostech také. Měření je nejvíce ovlivněno zavíráním dveří v měřicí místnosti i v místnostech okolo. Těmto chybám se nám nepodařilo zabránit.

### 9.2.2 Odhad doby měření

Doba měření je závislá především na počtu měřených bodů, délce kroku přiblížení v ose Z a vzdálenosti měřicího hrotu ve výchozí pozici od měřeného vzorku. Jelikož vzorek může být asymetrický (mít různou výšku v různých bodech), nelze dobu měření nijak přesně odhadnout. Při měření vzorku (dna nábojnice) 100x100 bodů, přibližovací krok v ose Z 2  $\mu\text{m}$  a vzdálenosti měřeného předmětu 1mm trvalo odměření 105 hodin. Průměrný čas jednoho měřeného bodu je tedy 38 sekund. Dle tohoto měření si můžete udělat velmi hrubou představu závislosti počtu měřených bodů na čase. Podotýkám však, že měřený čas na jeden bod se bude u každého měření lišit skrze změnu jednotlivých parametrů a tento odhad je čistě pro představu o časové náročnosti (viz Obrázek 23).

Při druhém měření byl čas jednoho měření 62 sekund. Vstupní parametry byly:

- Počet bodů - 100x100 bodů
- Přibližovací krok v ose Z 2  $\mu\text{m}$
- Vzdálenosti měřeného předmětu méně než 1mm



Obrázek 23 Vztah mezi časem a počtem měřených bodů

## 10 MĚŘENÍ VYBRANÝCH VZORKŮ A HODNOCENÍ VÝSLEDKŮ

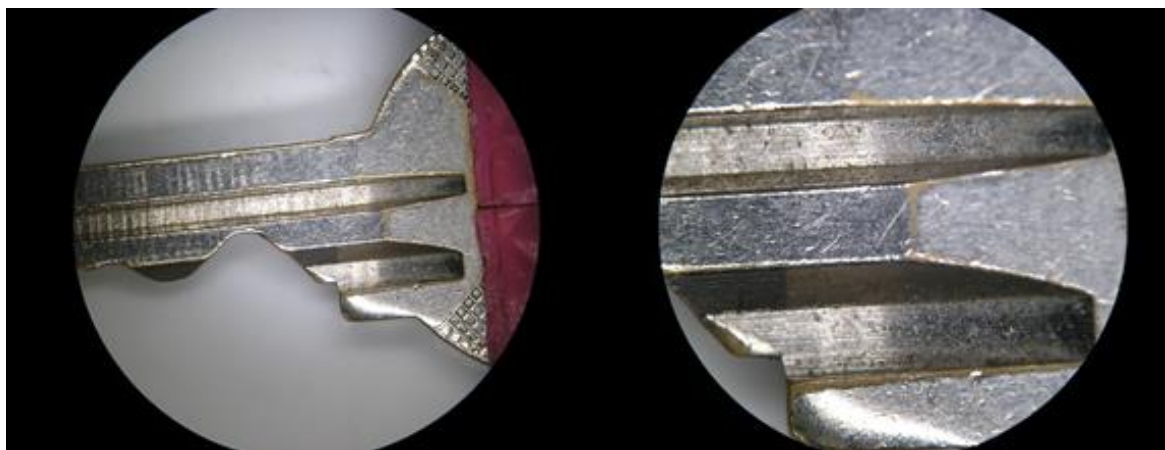
Všechny měřené vzorky byly vybírány kovové, protože námi použitá metoda je možná pouze s vodivými vzorky. Při měření se využívá fyzikálních principů měření procházejícího proudu. Vzorky jsem vybíral taky proto, aby vynikla měřicí metoda. Povrch jednotlivých vzorků se liší. Jak výškou měřeného vzorku, tak i členitostí povrchu. Na následujícím obrázku jsou fotky měřených vzorků pod stereomikroskopem.



Obrázek 24 Měřené vzorky – dno nábojnice, pětikoruna, klíč

### 10.1 Skenovaný vzorek - klíč

První měřený vzorek byl klíč, konkrétně drážky klíče. Měřený vzorek a jeho detail skenovaný na stereomikroskopu je na obrázku 25.



Obrázek 25 Klíč a jeho detail pod stereomikroskopem

Na obrázku 26 je graf zobrazující skenovaný povrch klíče. První obrázek je vykreslení plochy ve 3D. V okně Figure bylo zvolena mírně nakloněná perspektiva, aby šlo vidět i do drážek klíče. Nejlepší představu topografii povrchu však získáte, když sami otevřete měřený soubor a důkladně ho prozkoumáte, s použitím rotací náhledu a přiblížení (viz obrázek 26).

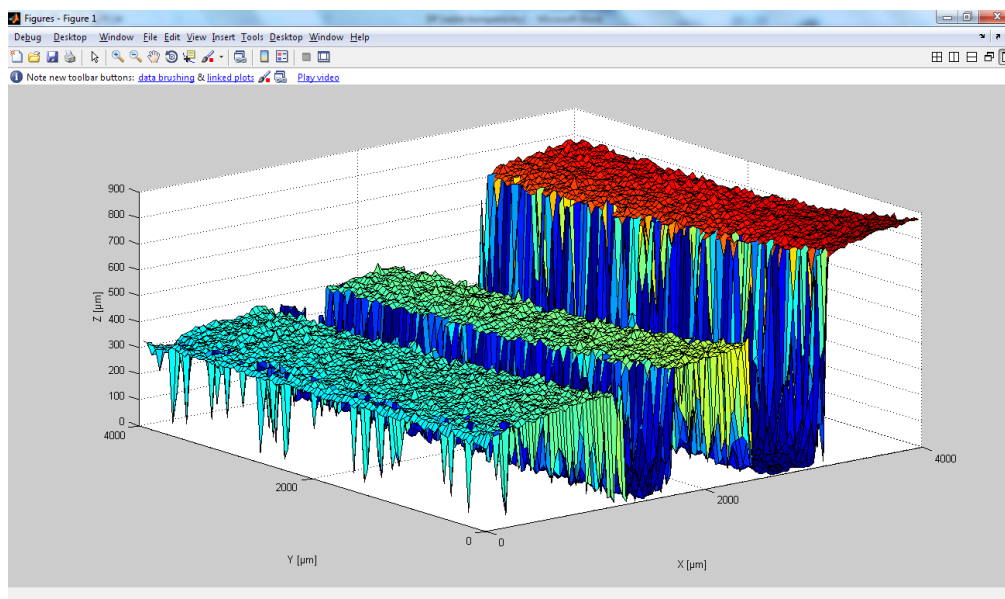
Vstupní hodnoty programu pro měření:

Délka kroku posuvu v ose X a Y - 40  $\mu\text{m}$

Přibližovací krok v ose Z - 2  $\mu\text{m}$

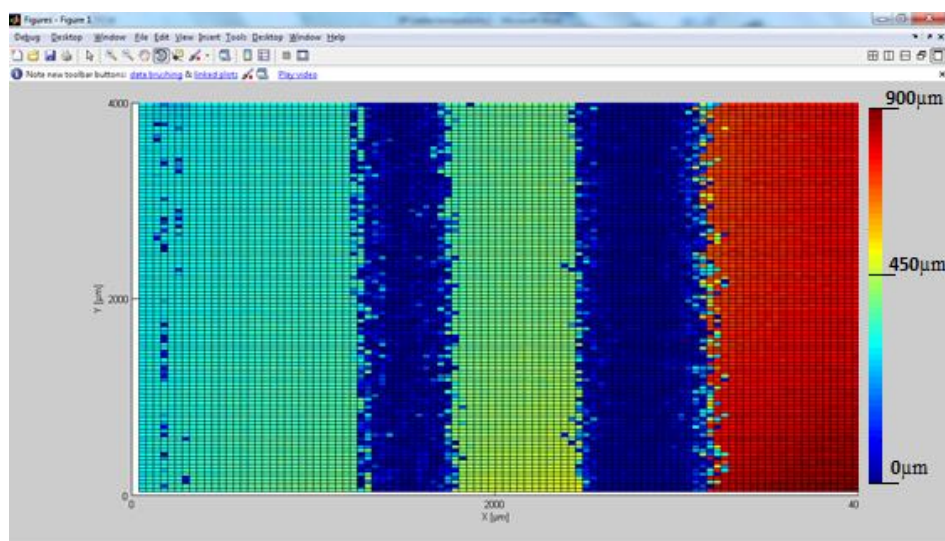
Počet měřených bodů - 10 000 (100x100 bodů)

Název - klic100.mat



Obrázek 26 Měřený vzorek – drážky klíče pohled 3D

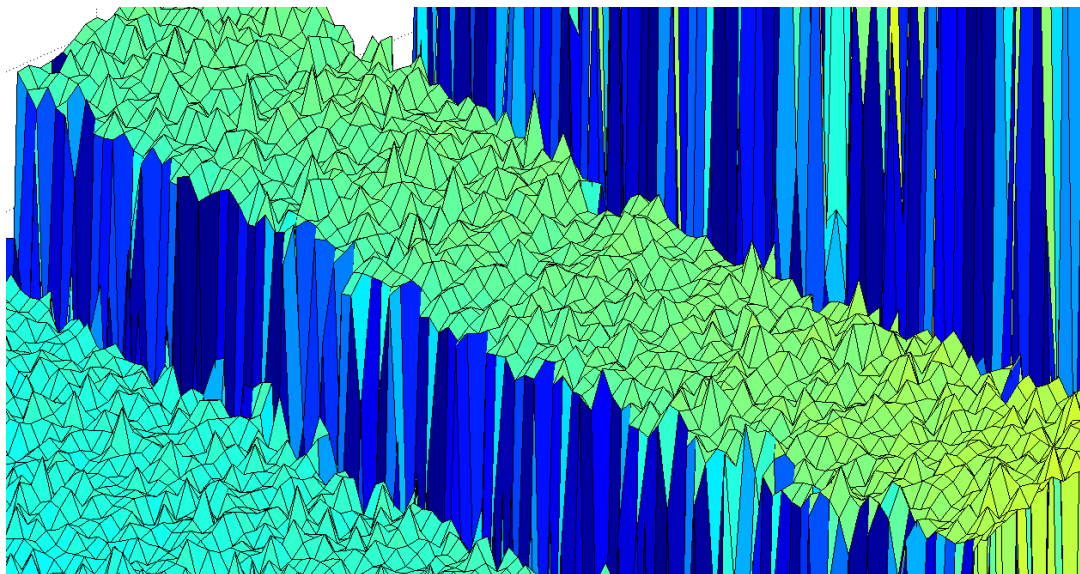
Další obrázek zobrazuje vykreslení drážek klíče z 2D pohledu. Měřenou hodnotu v ose Z můžeme odhadnout skrze zabarvení v měřeném bodu. Stupnice od 0 do 900  $\mu\text{m}$  nám dává představu o výšce skenované plochy (viz obrázek 27).



Obrázek 27 Měřený vzorek – drážky klíče, pohled 2D



Následující obrázek zobrazuje detailní pohled na měřený vzorek (viz obrázek 28).



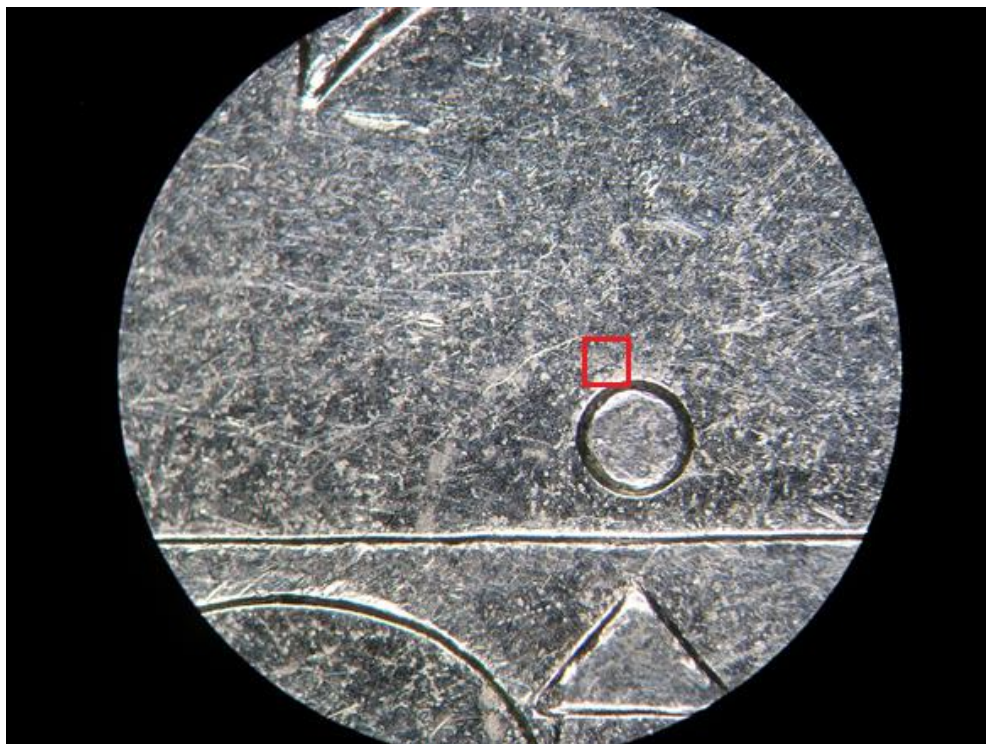
Obrázek 28 Měřený vzorek – drážky klíče, detail

## 10.2 Skenovaný vzorek - pětikoruna

Druhým měřeným vzorkem je plocha pětikoruny, nerovnosti na zdánlivě rovném povrchu. Zobrazení pod stereomikroskopem je na obrázku 29. Měřená plocha a její detail je zobrazena na obrázku 30.



Obrázek 29 Plocha pětikoruny pod stereomikroskopem



Obrázek 30 Měřená plocha pětikoruny pod stereomikroskopem - detail

Na obrázku obrázku 31 je zobrazena měřená plocha pětikoruny. Měřená plocha má rozměr 320x 320  $\mu\text{m}$ . Měřeným vzorkem byla zdánlivě rovná plocha na pětikoruně (viz obrázek 30).

Vstupní hodnoty programu pro měření:

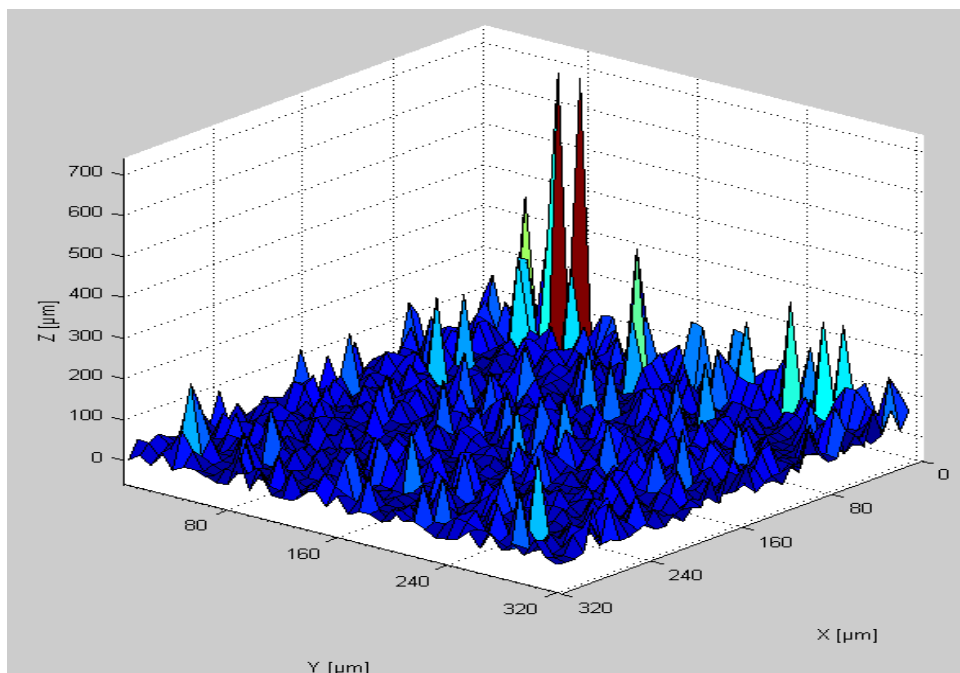
Délka kroku posuvu v ose X a Y - 8  $\mu\text{m}$

Přibližovací krok v ose Z - 30  $\mu\text{m}$

Počet měřených bodů - 1 600 (40x40 bodů)

Název - mince40.mat





Obrázek 31 Měřený vzorek – pětikoruna

### 10.3 Skenovaný vzorek - dno nábojnice

Třetí vzorek zkoumá plochu dna nábojnice (viz obrázek 32). Jeho detail a zobrazení měřené plochy je na obrázku 33.



Obrázek 32 Dno nábojnice pod stereomikroskopem



Obrázek 33 Měřená plocha dna nábojnice pod stereomikroskopem - detail

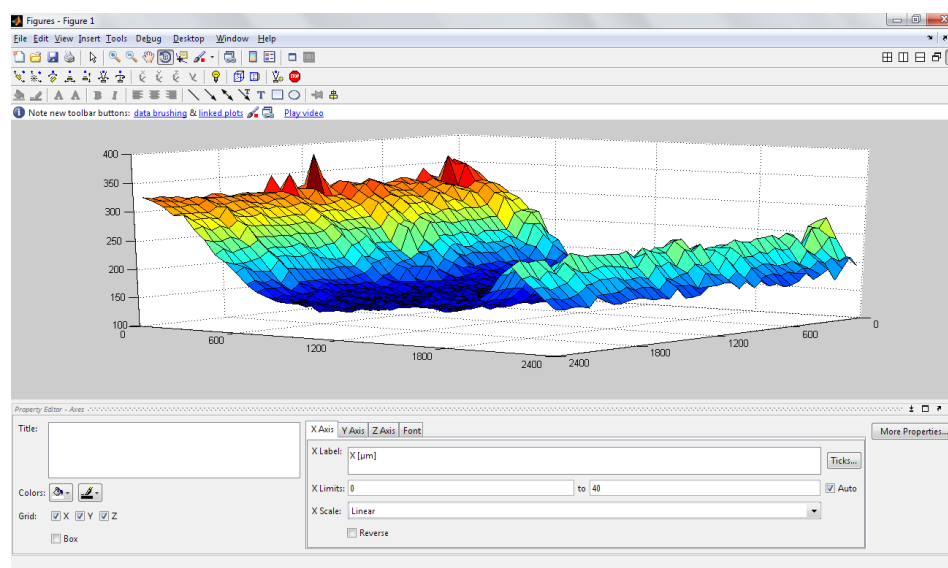
Detail zkoumá deformace dna nábojnice (viz obrázek 33). Obrázek 34 přibližuje i možnosti zobrazení v okně Figure. Můžete vidět nastavení názvu os, nastavení hodnot pro jednotlivé osy nebo volbu barevného nastavení zobrazení. Na dalších obrázcích jdou vidět možnosti nastavení v okně figure, možnosti rotace, změny perspektivy, popis os, změna hodnoty v popisu grafu, legenda a jiné (viz obrázek 34, 35).

Délka kroku posuvu v ose X a Y - 60  $\mu\text{m}$

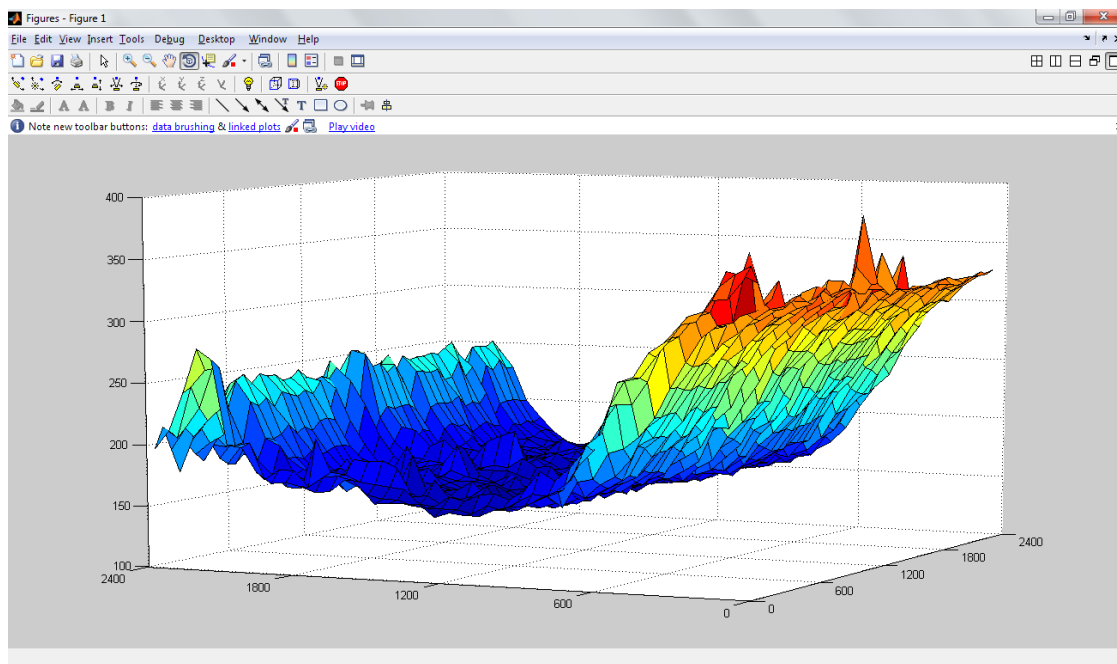
Přibližovací krok v ose Z - 5  $\mu\text{m}$

Počet měřených bodů - 1 600 (40x40 bodů)

Název - nabojnice40.mat



Obrázek 34 Měřený vzorek – nábojnice, deformace dna nábojnice pohled 1



Obrázek 35 Měřený vzorek – nábojnice, deformace dna nábojnice pohled 2

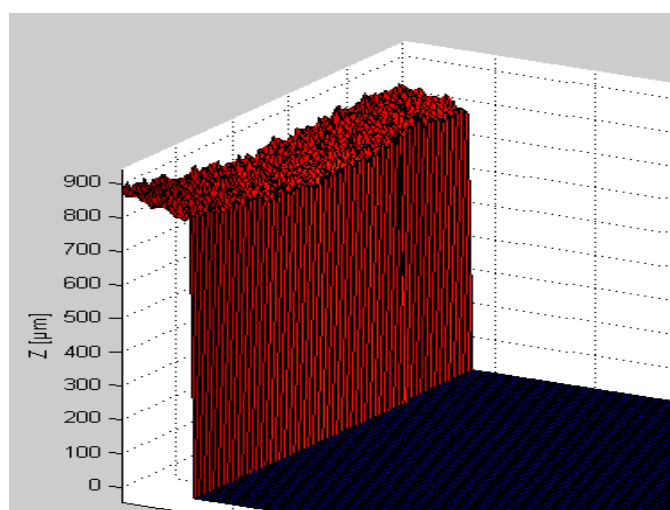
Další obrázek zobrazuje detail struktury měřené plochy v průběhu měření. Měření je právě v 20 %. Obrázek 37 zachycuje dynamické vykreslování hodnot.

Délka kroku posuvu v ose X a Y - 40  $\mu\text{m}$

Přibližovací krok v ose Z - 2  $\mu\text{m}$

Počet měřených bodů - 10 000 (100x100 bodů)

Název - mince100\_2.mat



Obrázek 36 Měřený vzorek – pětikoruna, proces dynamického vykreslování – detail

## ZÁVĚR

Po seznámení s programovacím prostředím MATLAB a s principy tvorby grafického uživatelského rozhraní byly definovány požadavky na GUI pro skenovací mikroskopii. Poté byly navrženy objekty a funkce, které utvářejí výsledný vzhled našeho rozhraní. Následovalo sestavení vývojového diagramu požadovaných funkcí programu a jejich návazností. Byl navržen způsob ukládání a vizualizace dat. Poté bylo GUI vytvořeno a naprogramováno pro účely měření. Byly ošetřeny jednotlivé chyby, program testován a následně vzniklé chyby byly opraveny. Finální verze programu nevykázala žádné chyby při sérii měření. Měřené vzorky pak byly vizualizovány a interpretovány. Celý tento proces byl hodnocen jako úspěšný, protože splnil zadané požadavky. Bezproblémové fungování je závislé na všech procesech, které proběhly. Od návrhu uživatelského rozhraní, přes návrh vývojového diagramu funkcí programu, který později snížil vznik chyb celého programu, až po způsob ukládání dat, jejich vizualizaci a interpretaci. Program prokázal úspěšnost bezproblémovým měřením, ukládáním dat, jejich vizualizací a také uživatelským rozhraním ošetřeným proti chybám, jak na straně uživatele, tak na straně programu.

Problémy, které při měření vznikaly v průběhu tvorby programu, byly průběžně ošetřeny. Problémy, které se nepodařilo eliminovat, jsou zejména časová náročnost měření na zastaralé výpočetní technice a vliv vibrací na měření.

Zastaralá výpočetní technika, na které se provádělo měření, se často stávala problémem. Při průběžném vykreslování grafu se doba měření zvýšila na dvojnásobek, a to teprve v 1/5 měřeného programu, kdy se do grafu načítala pouze pětina naměřených hodnot. To nás vedlo k vytvoření volby mezi možnostmi vykreslovat graf pouze po naměření celého řádku nebo graf vůbec nevykreslovat, pouze zapisovat hodnoty do souboru.

Zkreslení měření způsoboval vliv vibrací, změny tlaku v místnosti a jeho přilehlém okolí. Zejména mechanické vibrace, skrze činnost doktorandů v měřicí místnosti a přilehlých kancelářích. Mechanické vibrace počítačů, ventilátorů, klimatizace a jiného zařízení v místnosti, mechanické vibrace při zavírání dveří a změna tlaku v místnosti při otevírání, zavírání dveří způsobovaly zkreslení. Toto zkreslení se z časových důvodů nepodařilo určit, protože jedno měření plochy s 10 000 body trvalo přes čtyři dny.

Byly tu i problémy, kterým se zamezit podařilo. Například zvýšení přesnosti měření zvýšením počtu měřených bodů, snížením měřicí délky kroku pro posuv v ose Z a následné

zpřesnění měření. Výměna impedanční sondy za přesnější s ostrým hrotem a očištění zoxidovaných ploch měřeného vzorku. Největším problémem pro přesné měření se však stala časová náročnost pro jednotlivá měření. Měření vzorku v 10 000 bodech při měřicím kroku v ose Z o hodnotě 2  $\mu\text{m}$  trvalo 105 hodin. Proto následující měření musely být z časových důvodů zrychleny měřením méně bodů nebo s menší přesností.

I přes zmíněné vzniklé problémy byly naměřené hodnoty vykreslené grafem dostatečně přesné, aby přiblížily, jak daný skenovaný povrch vypadá. Skrze kvalitní vizualizaci programu MATLAB můžeme měřený povrch důkladně prozkoumat. Takto vizualizovaný graf můžeme dále upravovat, měnit perspektivu, přiblížit, rotovat, prozkoumat velmi přesně skenovanou plochu a to díky moderním přístrojům pro skenování, programovým a zobrazovacím technologiím.

## SEZNAM POUŽITÉ LITERATURY

- [1] MATLAB – Wikipedie. *Wikipedie, otevřená encyklopedie* [online]. 2014 [cit. 2014-02-20]. Dostupné z: <http://cs.wikipedia.org/wiki/MATLAB>
- [2] MATLAB Programming Language. MATLAB Programming Language, Programming with MATLAB, Programming Language MATLAB, Learn MATLAB Online. [online]. 2014 [cit. 2014-03-12]. Dostupné z: <http://www.altiusdirectory.com/Computers/MATLAB-programming-language.php>
- [3] *Úvod do používání MATLAB*. Univerzita Pardubice, 1997. Učební text. Univerzita Pardubice. Vedoucí práce Ing. Dušek František, CSc.
- [4] Grafické uživatelské rozhraní – Wikipedie. In: *Wikipedia: the free encyclopedia* [online]. San Francisco (CA): Wikimedia Foundation, 2001- [cit. 2014-03-19]. Dostupné z: [http://cs.wikipedia.org/wiki/Grafick%C3%A9\\_u%C5%BEivatelsk%C3%A9\\_rozhran%C3%AD](http://cs.wikipedia.org/wiki/Grafick%C3%A9_u%C5%BEivatelsk%C3%A9_rozhran%C3%AD)
- [5] DOSTÁL, Martin. Základy tvorby uživatelského rozhraní. Olomouc, 2010. Dostupné z: <http://dostal.inf.upol.cz/data/0910/URO/uro-18-12-2010.pdf>. Výukový text. Univerzita Palackého v Olomouc.
- [6] ZAPLATÍLEK, Karel a Bohuslav DOŇAR. MATLAB: tvorba uživatelských aplikací. 1. vyd. Praha, 2004, 215 s. ISBN 80-730-0133-0.
- [7] The GNOME Project. GNOME Human Interface GUIDelines[online] - [cit. 2014-19-03] , dostupný z: <http://library.gnome.org/devel/hig-book/stable/>
- [8] MATLAB: the language of technical computing [online]. Massachusetts: The MathWorks, Inc., 1999, 1 sv. (různé číslování stránek). [cit. 2014-04-01]. ISBN The Language of Technical Computing. Dostupné z: <https://courses.cit.cornell.edu/bionb442/GUIDesign/BUILDGUI.PDF>
- [9] MANOVICH, Lev. Articles -Interaction as an Aesthetic Event[online] 2007 - [cit.2014-3-05] , dostupný z: [http://www.manovich.net/DOCS/TATE\\_lecture.doc](http://www.manovich.net/DOCS/TATE_lecture.doc)
- [10] ČAPEK, Miroslav. Uživatelské aplikace: návrh GUI části programu v MATLABu. Uživatelské aplikace: návrh GUI části programu v MATLABu [online]. 2010, č. 1

- [cit. 2014-05-02]. Dostupné z:  
[http://www.old.elmag.org/lib/exe/fetch.php/wiki:user:capek:maa\\_pr12\\_gui.pdf](http://www.old.elmag.org/lib/exe/fetch.php/wiki:user:capek:maa_pr12_gui.pdf)
- [11] MACHALA, Libor, et al. Mikroskopie skenující sondou [online]. Olomouc : Univerzita Palackého Olomouc, 2003 [cit. 2011-02-23]. Dostupné z WWW:<http://atmilab.upol.cz/mss/>>.
- [12] Laboratoř mikroskopie atomárních sil [online]. [200?] [cit. 2011-02-23]. Mikroskopie skenující sondou. Dostupné z WWW: <<http://atmilab.upol.cz/spm.html>>.
- [13] KUBÍNEK, Roman. Rastrovací sondová mikroskopie [online]. [s.l.] : Katedra experimentální fyziky Přírodovědecké fakulty v Olomouci, 2000 [cit. 2011-03-01]. Dostupné z WWW: <[http://exfyz.upol.cz/bf/predn/exp\\_met1/rsm/](http://exfyz.upol.cz/bf/predn/exp_met1/rsm/)>.
- [14] KUDĚLKA, Josef. Metrologické zabezpečení mikronových a submikronových posuvů: Metrological problems in micron and submicron distances. Zlín, 2011. Diplomová práce. Univerzita Tomáše Bati. Vedoucí práce doc.RNDr. Vojtěch Křesálek, CSc.
- [15] KUDĚLKA, Josef; MARTÍNEK, Tomáš. Studium a vytváření struktur v submikrometrové oblasti. Studentská tvůrčí a odborná činnost. Zlín : Univerzita Tomáše Bati ve Zlíně, Fakulta aplikované informatiky, 2010. s. 14.
- [16] Users GUIDE. Agilent Technologies. Agilent 34401A 6 Digit Multimeter [online]. Seventh edition. Santa Clara, 2007 [cit. 2011-02-15]. Dostupné z WWW: <http://www.home.agilent.com/agilent/product.jspx?cc=CZ&lc=end&nid=-536902435.536880933&pageMode=PL>.
- [17] Product Overview. Agilent Technologies. Agilent 34401A Multimeter [online]. 2007 [cit. 2011-02-15]. Dostupné z WWW: <<http://www.home.agilent.com/agilent/product.jspx?cc=CZ&lc=end&nid=-536902435.536880933&pageMode=PL>>.
- [18] Service GUIDE. Agilent Technologies. Agilent 34401A 6½ Digit Multimeter [online]. Seventh edition. Santa Clara, 2007 [cit. 2011-02-15]. Dostupné z WWW:<<http://www.home.agilent.com/agilent/product.jspx?cc=CZ&lc=end&nid=-536902435.536880933&pageMode=PL>>.

- [19] MS 74E User Manual. Physik Instrumente. Mercury C-862 Networkable Single-Axis DC-Motor Controller. Release 8.41. Karlsruhe, 2004



**SEZNAM POUŽITÝCH SYMBOLŮ A ZKRATEK**

CLI	Command Line
CNC	Computer Numerc Control
DC	Direct Current
GUI	Graphical User Interface
GUIDE	Graphical User Interface Development Environment
GPIB	General Purpose Interface Bus
HDD	Hard Disc Drive
HUD	Head Up Display
MATLAB	Matrix Laboratory
PARC	Palo Alto Research Center
PC	Personal Computer
SMP	Scanning Probe Microscopy
STM	Scanning Tunneling Microscopy
WIMP	Window, Icon, Menu, Pointing device

**SEZNAM OBRÁZKŮ**

Obrázek 1 Proces návrhu vzhledu GUI [8].....	22
Obrázek 2 Přehled ui-objektů [10].....	24
Obrázek 3 Přehled uicontrol prvků [10] .....	25
Obrázek 4 Skenovací tunelovací mikroskopie [11] .....	32
Obrázek 5 Režim s konstantní výškou a konstantním proudem [13] .....	33
Obrázek 6 Fotografie STM hrotu [11] .....	34
Obrázek 7 Nejbližší přiblížení vlivem nenulové šířky hrotu [11] .....	35
Obrázek 8 Mercury M-110 1DG ve srovnání s 9V baterií [15].....	36
Obrázek 9 Mercury C-862 ve srovnání s mincemi 1\$ a 1€ [19].....	37
Obrázek 10 Přehled provedených měření a přehled naměřených hodnot[15].....	38
Obrázek 11 Přední panel digitálního multimetru Hewlett Packard 34401A [17].....	39
Obrázek 12 P-611.3S NanoCube [14] .....	40
Obrázek 13 P-611.3 NanoCube v kombinaci se soustavou M-111 1DG [14].....	40
Obrázek 14 Program MATLAB, základní okna a jejich popisky.....	44
Obrázek 15 Grafické znázornění jednotlivých uicontrol prvků.....	44
Obrázek 16 GUIDE – nástroj pro tvorbu uživatelského rozhraní .....	45
Obrázek 17 Dávkový soubor (m-soubor) .....	46
Obrázek 18 Zobrazení ve figure .....	46
Obrázek 19 Výsledný vzhled GUI.....	49
Obrázek 20 Funkce waitbar pro znázornění průběhu měření .....	51
Obrázek 21 Schéma algoritmu programu .....	52
Obrázek 22 Struktury ukládaných hodnot do matice v programu MATLAB .....	56
Obrázek 23 Vztah mezi časem a počtem měřených bodů .....	60
Obrázek 24 Měřené vzorky – dno nábojnice, pětikoruna, klíč.....	61
Obrázek 25 Klíč a jeho detail pod stereomikroskopem.....	61
Obrázek 26 Měřený vzorek – drážky klíče pohled 3D .....	62
Obrázek 27 Měřený vzorek – drážky klíče, pohled 2D .....	62
Obrázek 28 Měřený vzorek – drážky klíče, detail .....	63
Obrázek 29 Plocha pětikoruny pod stereomikroskopem .....	63
Obrázek 30 Měřená plocha pětikoruny pod stereomikroskopem - detail .....	64
Obrázek 31 Měřený vzorek – pětikoruna .....	65
Obrázek 32 Dno nábojnice pod stereomikroskopem.....	65

Obrázek 33 Měřená plocha dna nábojnice pod stereomikroskopem - detail .....	66
Obrázek 34 Měřený vzorek – nábojnice, deformace dna nábojnice pohled 1 .....	66
Obrázek 35 Měřený vzorek – nábojnice, deformace dna nábojnice pohled 2 .....	67
Obrázek 36 Měřený vzorek – pětikoruna, proces dynamického vykreslování – detail .....	67

**SEZNAM TABULEK**

Tabulka 1 Indexace při zápisu do matice.....	55
--	----

## SEZNAM PŘÍLOH

Zdrojový kód programu:

GUI\_mereni\_mikroposuvu.m

GUI\_mereni\_mikroposuvu.fig

GUI\_mereni\_mikroposuvu.asv

Fotky:

Stereomikroskop1 - klíč

Stereomikroskop2 - klíč\_detail

Stereomikroskop3 - pětikoruna

Stereomikroskop4 - pětikoruna\_detail

Stereomikroskop5 - dno nábojnice

Stereomikroskop6 - dno nábojnice\_detail

Měřené vzorky:

klic100.mat

mince40.mat

nabojnice40.mat

mince100\_2.mat