

Rozpoznávání tvarů

Projekt do předmětu Umělá inteligence

Obsah

Zadání.....	1
Princip rozpoznávání	2
Odhad dle poměru stran.....	2
Plán na vylepšení	2
Implementace	3
Podpora prohlížečů.....	3
Rozpoznávací jádro.....	3
Ukázka rozpoznání tvaru.....	4
Implementované tvary	5
Obdélník.....	5
Kosočtverec.....	5
Kružnice	5
Rozšíření.....	6
Spojovací čáry.....	6
Označení objektu	6
Označení bloku.....	6
Odznačení bloku	6
Budoucnost	7
Editace popisků	7
Metadata	7
Výrazy	7

Zadání

Vytvořte systém pro rozpoznávání tvaru nakreslených myší nebo tabletem na obrazovku.

Požadavky:

- programovací jazyk: JavaScript / JScript (rozpoznávání se provádí ve webové aplikaci na straně klienta)
- musí umět rozpoznat (části vývojového diagramu):
 - obdélník (akce)
 - kruh (mezi-stav)
 - kosočtverec (větvení)
 - spojovací čáry
- volitelné rozpoznání:
 - tažení objektu
 - označení bloku
 - odznačení
- ne příliš složitá rozšiřitelnost o další tvary

Princip rozpoznávání

Rozpoznání tvaru sestává z následovních kroků:

- o snímání vstupu
- o spočtení minima a maxima souřadnic
- o spočtení rozměrů
- o redukce možných tvarů podle poměru stran
- o výběr tvaru dle nejmenší směrodatné odchylky od referenčního tvaru

Odhad dle poměru stran

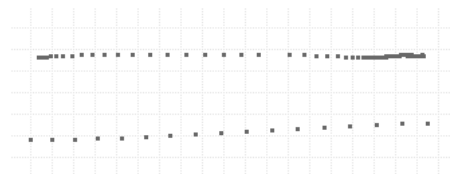
Aby se zamezilo špatné detekci, před počítáním odchylek se provede filtrování tvarů dle poměru stran. Největší přínos má použití této dodatečné funkce u tvarů jako jsou svislá nebo vodorovná čára a kružnice. Podmínky, které se kladou na tyto tvary jsou konkrétně:

```
function shape_size_check( shape, x ) { // x = width / height
  switch(shape) {
    case 'circle' :
      return (x>0.66) && (x<=1.5); // not too ellipsoid
    case 'vertical' :
      return (x<=0.25); // a vertical line
    case 'horizontal' :
      return (x>=4); // a horizontal line
  }
  return true;
}
```

Tyto podmínky říkají, že co nemá jeden rozměr alespoň čtyřikrát delší než druhý nemůže být vodorovná nebo svislá čára. U kružnice se omezuje na poměr stran 3:2 - co je placatější už není kružnice ale elipsa (a tu teď nechceme rozpoznávat).

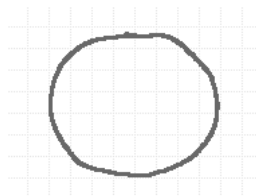
Plán na vylepšení

Během práce byl objevený neduh způsobený nestejným snímáním bodů. Při rychlém pohybu jsou body vzdáleny více než při pomalém pohybu. Pravděpodobně to je způsobeno konstantní vzorkovací frekvencí, takže s tím přímo nejde nic udělat, aby se to snímalo rovnoměrně.



Obr. 1: Nestejně snímání (nahoru) a ideální vstup (dolů)

Nápad provést předzpracování vyplynul ale také z toho, že u dlouhých křivek je bodů hodně a rozpoznání je ztlačně pomalejší - takže se požaduje snížení počtu vstupních bodů na několiknásobek počtu referenčních bodů (namísto desítek až stovek).



Obr. 2: Příliš mnoho vstupních bodů

Řešením by mohlo být předzpracování bodů tak, že se tvar křivky zachová ale body budou na ní umístěny více rovnoměrně - vypuštěním zbytečných a doplněním interpolovaných pozic bodů.

Implementace

Projekt byl implementován pomocí *JavaScriptu* (v prohlížeči *Internet Explorer* se obdoba technologie jmenuje *JScript*) a vyžaduje od prohlížeče podporu objektového modelu - *DOM* (*Document Object Model*) a kaskádových stylů - *CSS* (*Cascading Style Sheets*).

Podpora prohlížečů

Podporované prohlížeče jsou:

- *Mozilla Firefox*
(primární cíl, testováno na verzi 1.0+ pro Windows a 1.0.3 pro Linux)
- *Microsoft Internet Explorer 6.0*
(kód pro vykreslování byl upraven pro odlišný box-model a průhlednost řešena jinak)

Nepodporované prohlížeče jsou:

- *Opera 8*
(výstup je zkruslený - chybějící podpora vlastnosti *opacity* z doporučení *CSS* verze 3)
- všechny ostatní

Podpora pro prohlížeč *Opera 8* bude možná v případě, že tento prohlížeč implementuje uvedenou vlastnost z doporučení *CSS3*.

Rozpoznávací jádro

Rozhraní jádra pro rozpoznávání tvarů obsahuje jednu metodu a to:

```
function resolve_object( paths, last )
```

Do této funkce s booleovskou návratovou hodnotou vstupuje pole tahů a indikace, jestli je tah poslední (tj. uživatel už nenakreslil žádný další tah v rámci určitého časového intervalu - v ukázce je použita hodnota 500ms).

Funkce pak vrací jestli byl rozpoznán nějaký tvar, takže třeba vstupní funkce která zachytává body může smazat dočasnou vizualizaci tahů (v ukázce malé tečky, resp. čtverce veliké 3x3 pixelů).

Pokud se nějaký tvar rozpozná, tak je volána další funkce (z vyšší vrstvy):

```
function shape_detected( shape, x, y, a, b, last )
```

Tato funkce zabezpečuje nejen zpracování tvaru, ale může taky radit (*hint*) rozpoznávacímu jádru. Funkce musí vrátit *true*, pokud je tvar konečný a zpracovaný, nebo *false* pokud se použila rada (a přestavila se množina rozpoznatelných tvarů). V druhém případě vrátí funkce resolve_object *false*, tj. je uživateli umožněno dokreslit symbol dalším tahem. V projektu je použito tohle jenom u kreslení obdélníku dvěma tahy tak, že se nejprve nakreslí svislá čára:

```
// multiple stroke shapes
if (!last) {

    // hinting
    if ( (shape=='vertical') || (shape=='horizontal') ) {
        shapes = [ 'rectangle' ];
        return false;
    }

}
```

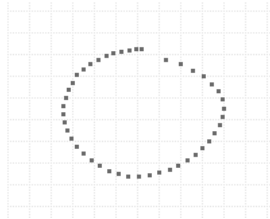
Zbytek modulu pro rozpoznávání sestává z funkce pro kontrolu dle poměru stran, funkce pro výpočet referenčních bodů a funkce pro spočtení směrodatné odchylky od daného tvaru:

```
function shape_size_check( shape, x, y, a, b ) // x = width / height
function shape_generate ( shape, x, y, a, b )
function shape_similarity( cp, inp )
```

Rozšíření o další tvary se provede doplněním kódu pro do prvních dvou funkcí.

Ukázka rozpoznání tvaru

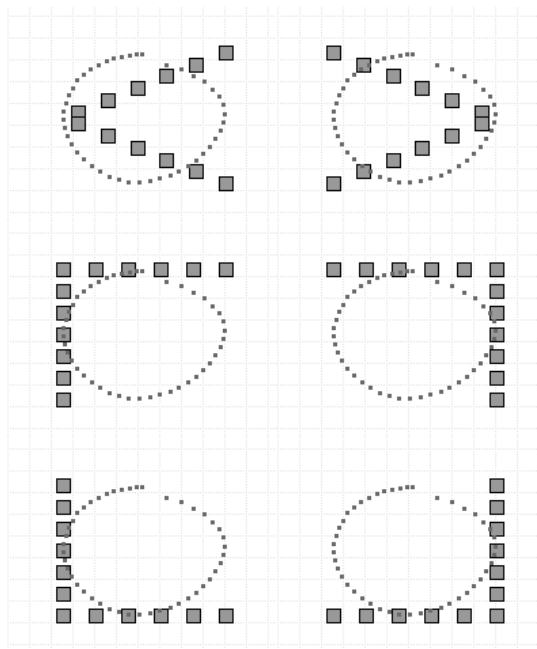
Vstupní posloupnost bodů tvoří kruh, nebo alespoň úmyslem bylo nakreslit kruh:



Obr. 3: Vstupní data

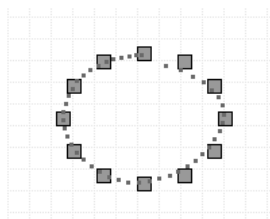
Z dat se zjistí minimální a maximální souřadnice, z nich se odvodí velikost, a z velikostí dostáváme poměr stran. Pomocí tohoto poměru se odfiltrují některé obrazce (konkrétně jde o obdélník a kosočtverec, protože poměr stran se přibližuje k jedničce, ale tyto dva obrazy mají poměr stran x/y kolem dvou).

Po filtraci tvarů pomocí poměru stran se dostáváme k vlastnímu porovnání:



Obr. 4: Obrazce LT, GT, WS, ES, SE, SW

Nejmenší odchylku mají ale body od vstupního obrazce typu CIRCLE, což je kruh, takže se z funkce pro rozpoznání vrátí právě tento tvar. Grafické znázornění vstupních bodů a bodů ideálního obrazce je zde:

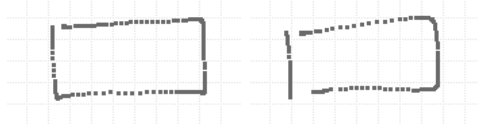


Obr. 5: Obrazec CIRCLE

Implementované tvary

Zde jsou uvedeny tvary které se snažím rozpoznávat - pokud je víc možností jak daný tvar zadat, tak jsou uvedeny všechny.

Obdélník



Obr. 6: Úplné tvary pomocí jednoho nebo dvou tahů



Obr. 7: Zkrácené tvary pro urychlení práce

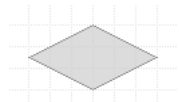


Obr. 8: Zobrazený obdélník

Kosočtverec

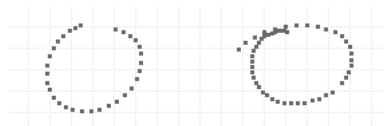


Obr. 9: Úplný tvar a dva zkrácené pro urychlení práce



Obr. 10: Zobrazený kosočtverec

Kružnice



Obr. 11: Úplné tvary pro kružnici - rozpozná se i přetažení



Obr. 12: Zobrazená kružnice

Rozšíření

Spojovací čáry

Spojovací čáry nejsou ještě implementovány, protože rozpoznané objekty nejsou vlastně reprezentovány objekty dle toho co znamenají, ale jsou jenom geometrické tvary nakreslené na obrazovku.

Až se bude implementovat objektový model vývojového diagramu, tak několik položek bude reprezentováno i graficky. Tyto grafické objekty pak budou mít funkci na zjištění, jestli uživatel klikl uvnitř nebo vně zobrazeného tvaru.

Pak jestliže začalo tažení z jednoho objektu a skončilo ve druhém, je to 100% spojovací čára. V reálu ale tažení nebude muset končit ve druhém objektu, ale může před ním, takže se musí provést odhad, ve kterém objektu mohla čára končit.

Pokud teda známe dva objekty které mají být spojeny čarou, tak můžeme přistoupit na rozpoznání čar, které bude pracovat ve čtyřech (90° úhel), šesti (60° úhel) nebo osmi směrech (45° úhel). Rozpoznávací algoritmus pro čáru ale bude jiný, protože z jedné posloupnosti vstupních bodů bude generovat několik segmentů čáry.

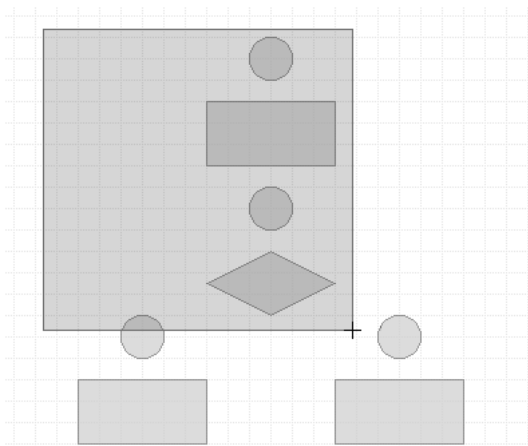
Označení objektu

Není implementováno, bude se provádět kliknutím na objekt. Pokud bude objekt označený, tak ho lze dalším tahem přesunout - pokud nebude, tah se bere jako počátek spojovací čáry nebo jiného vstupu určeného na rozpoznání.

Označení bloku

Je jediné sice jen částečně implementované rozšíření. Označení se provádí podržením pera ve výchozí pozici (resp. v okruhu daném tolerancí, nastaveno na 10px) po nějakou dobu a pak tažením. Doba nutná k přepnutí do označovacího režimu je nastavena na relativně krátký interval (200ms) protože pokud chce uživatel psát, tak rozhodně nebude držet pero v jedné pozici.

Moje částečná implementace neobsahuje práci s vlastními objekty, ale jenom implementaci přepnutí na režim označování bloku a pak grafické znázornění výběrového obdélníku následovně:



Obr. 13: Označení bloku

Odznačení bloku

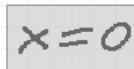
Odznačení se bude provádět na základě kliknutí do libovolné pozice (mimo nakreslené objekty) pokud předtím něco označeného bylo. Pokud označené nic nebylo, může se provést původní funkce kliknutí, tj. třeba opakované vložení posledního rozpoznávaného objektu nebo obdobná akce ulehčující práci návrháře.

Budoucnost

Tato sekce obsahuje několik ukávek použití rozpoznávání vstupu v budoucnu - tj. to co mám v plánu implementovat do systému chystaného v méj diplomové práci.

Editace popisků

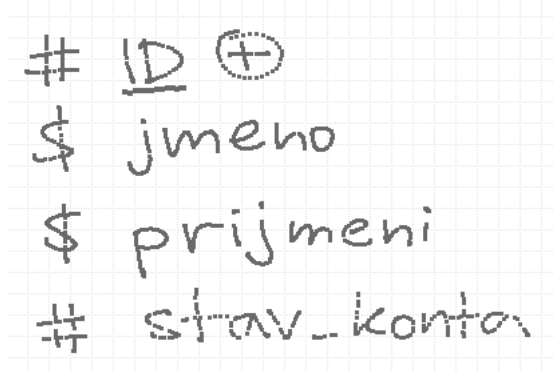
Pokud se dodělá rozpoznávání písmen a číslic, bude možné psát popisky přímo do tvarů:



Obr. 14: Vstup pro rozpoznání popisků

Metadata

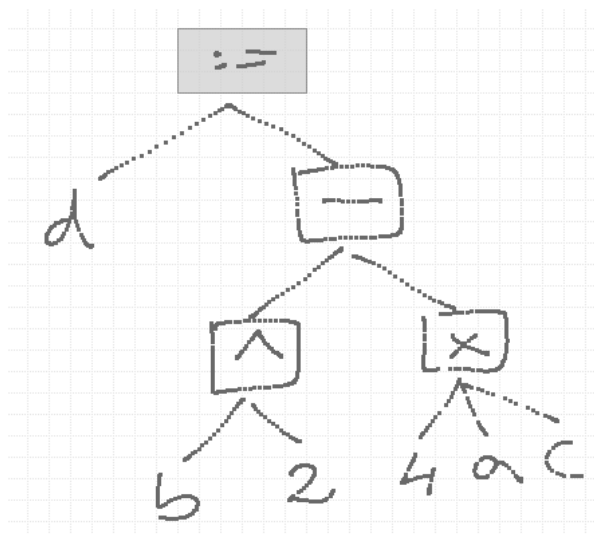
Rozpoznávat se dá nejenom písmo a tvar, ale také ostatní symboly - zde se plánuje použít rozpoznání pro typ sloupce, příznak auto-inkrementu (plus v kroužku) a primární klíč (podtržení):



Obr. 15: Vstup pro rozpoznání metadat

Výrazy

Výrazy bude možno zadat buď matematicky, vzorcem, nebo také sémantickým stromem:



Obr. 16: Vstup pro rozpoznání výrazů