|  |
| --- |
| SLOVENSKÁ TECHNICKÁ UNIVERZITA, FAKULTA ELEKTROTECHNIKY A INFORMATIKY |
| VIZS – úloha 1 |
| Návrh optickej odometrie pomocou optického toku |

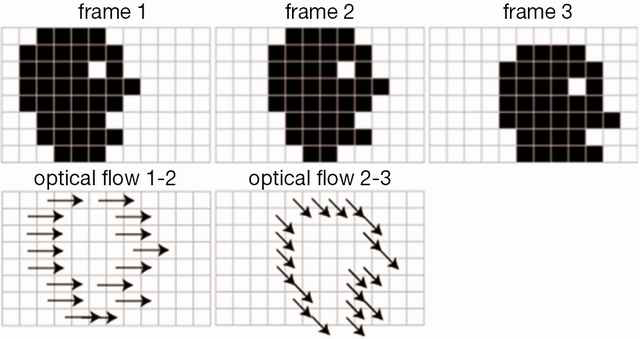
|  |
| --- |
| Adam Sojka 72515, Filip Štec 72520  9.4.2017 |

# Úloha

# Teoretický postup

## Optický tok

Pod pojmom optický tok rozumieme vizuálny pohyb predmetov na obrázkoch zachytených práve za sebou pomocou kamery. Pohyb pritom môže byť vyvolaný pohybom objektov vo videu alebo aj samotným pohybom kamery. Tento pohyb je zaznamenaný pomocou vektorov, ktoré spájajú určitý bod z predchádzajúceho záberu s tým istým bodom v aktuálnom zábere.

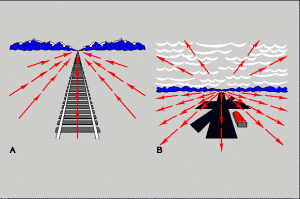


Obrázok 1 Prinícp určenie optického toku

Výpočet optického toku po jednotlivých pixeloch nie je možný, lebo sa dostávame do jednej rovnice s dvomi neznámymi, preto na výpočet použijeme Lucas-Kanade metódu. Táto metóda pozostáva z predpokladu, že susedné body sa pohybujú podobne ako vybratý, tým dostávame preurčenú sústavu rovníc. Metóda rieši aj problém s malými a veľkými pohybmi, lebo používa pyramídu. Pohybom hore po pyramíde sa odstraňujú malé pohyby a veľké pohyby sa stávajú malými, čím dostávame aj mierku.

## Vizuálna odometria

Najprv sa budeme zaoberať jednoduchšou úlohou, a to rozpoznanie smeru pohybu v osiach x, y a z. Pre os x, horizontálny pohyb s kamerou do strany a os z, vertikálny pohyb, je identifikácia smeru pohybu jednoduchá, keďže vektory sa pohybujú vždy iba jedným smerom, horizontálne alebo vertikálne. Pre pohyb v smere osi y, to je pohyb s kamerou dopredu a dozadu, je optický tok zobrazený na nasledujúcom obrázku. Ako vidno, pri pohybe dopredu vektory optického toku smerujú akoby z fotky a zas pri cúvaní idú do fotky. To by sa malo dať zistiť porovnávaním uhla natočenia vektora s jeho absolútnou pozíciou na fotke, keďže vieme predpokladať, že napr. v pravom dolnom rohu fotky pri pohybe dopredu by mal vektor mať uhol približne 45 stupňov a smerovať vpravo dole.



Obrázok 2 Princíp určenia pohybu v smere osi Y

Iným spôsobom určenia pohybu kamery v osi y je pomocou veľkosti detegovaného objektu na obrázku. Je potrebné najprv určiť závislosť medzi vzdialenosťou a veľkosťou objektu z dvoch obrázkov, kde vieme v akej vzdialenosti sa objekt nachádza od kamery a poznáme jeho skutočnú veľkosť aj veľkosť na obrázku.



Obrázok 3 Polomer kruhu sa so zväčšujúcou sa vzdialenosťou zmenšuje

Podľa určenej závislosti a zmeranej veľkosti objektu v pixeloch potom vieme určiť skutočnú vzdialenosť objektu. Závislosť je:

Kde Y je reálna vzdialenosť v centimetroch, k je smernica priamky, q je posunutie priamky a x je veľkosť pixelu v centimetroch:

Kde Xcm je veľkosť objektu v centimetroch a X je veľkosť objektu v pixeloch. Z uvedených vzťahov vyplýva, že vzdialenosť objektu od kamery závisí nepriamoúmerne od veľkosti objektu. Riešením sústavy rovníc o dvoch neznámych získame koeficienty rovnice priamky k a q.

# Programové vybavenie

Ku zdrojovým kódom ako aj tejto dokumentácii je možné sa dostať na umiestnení: <https://github.com/Smadas/VIZSul1>

Všetky zdrojové kódy sú písané v C++ s použitím openCv 3.1, aj staršie verzie by mali byť kompatibilné.

Programové vybavenie sa skladá z nasledovných knižníc:

1. Main – spúšťanie programu a všetkých podprogramov, hlavná slučka, ktorá prebieha všetky obrazy
2. captureVideo – záznam obrázkov z kamery pripojenej k počítaču a ich uloženie na disk
3. readImagesFromDirectory – načítanie všetkých obrázkov z umiestnenia na disku do pamäti RAM
4. DetectTarget – detekcia objektu záujmu
5. opticalFlowLucasKanade – výpočet optického toku
6. detectMotionXZ – detekcia pohybu kamery v osi X a Z
7. detectMotionY – detekcia pohybu kamery v osi Y
8. displayData – grafické a číselné zobrazenie pohybu kamery

## 1. Funkcia main

## 2. Záznam obrázkov z kamery

Obsahuje funkcie saveFrame(uloženie obrázka) a captureVideoAsImages(záznam videa ako obrázkov).

saveFrame – uloží cv maticu do bitovej mapy na disk do umiestnenia captureVid, ktoré je umiestnené pri spustiteľnom súbore.

captureVideoAsImages – zobrazuje obrázky z kamery do okna a spúšťa funkciu saveFrame každých 100ms.

## 3. Načítanie obrázkov do pamäte RAM

Obsahuje tri funkcie:

* getImageFileNames – načítanie názvov všetkých bitových máp v danom umiestnení, vstupom je umiestnenie súborov a výstupom je vektor reťazcov názvov.
* readImgFiles – načítanie bitových máp zo súborov do pamäte RAM, vstupom je umiestnenie obrázkov a výstupom je vektor cv matíc načítaných obrázkov.
* showImagesFromDirectory – demo zobrazenie obrázkov z daného umiestnenia

## 4. Detekcia objektu záujmu

## 5. Výpočet optického toku

## 6. Detekcia pohybu v osi XZ

## 7. Detekcia pohybu v osi Y

## 8. Zobrazenie pohybu kamery

# Výsledok odometrie

## Posun kamery od pozadia v osi Y





## Posun kamery v osi X

## Posun kamery v osi Z

## Posun kamery v osi X a súčasne Z

# Zdroje

<http://docs.opencv.org/2.4/doc/tutorials/introduction/windows_install/windows_install.html>

<http://docs.opencv.org/2.4/doc/tutorials/introduction/windows_visual_studio_Opencv/windows_visual_studio_Opencv.html>

<https://github.com/opencv/opencv/blob/master/samples/cpp/lkdemo.cpp>