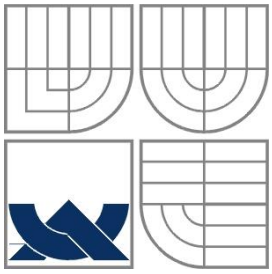
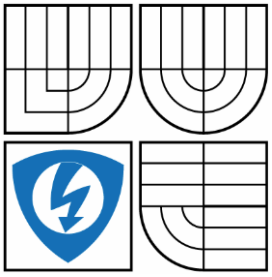


VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ

BRNO UNIVERSITY OF TECHNOLOGY



FAKULTA ELEKTROTECHNIKY A KOMUNIKAČNÍCH TECHNOLOGIÍ
ÚSTAV AUTOMATIZACE A MĚŘICÍ TECHNIKY

FACULTY OF ELECTRICAL ENGINEERING AND COMMUNICATION
DEPARTMENT OF CONTROL AND INSTRUMENTATION

TRASOVÁNÍ VÝZNAMNÝCH BODŮ VE VIDEOSEKVENCI NESTACIONÁRNÍ KAMERY

Interest Points Tracking in Video Sequence of Non-stationary Camera

DIPLOMOVÁ PRÁCE

AUTOR PRÁCE
AUTHOR

Bc. PAVEL STUDENÝ

VEDOUCÍ PRÁCE
SUPERVISOR

Ing. Karel Horák, Ph.D.

BRNO 2016

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
FAKULTA ELEKTROTECHNIKY A KOMUNIKAČNÍCH TECHNOLOGIÍ

Diplomová práce

magisterský navazující studijní obor **Kybernetika, automatizace a měření** Ústav automatizace a měřicí techniky

Student: Bc. Pavel Studený ***ID:*** 147388

Ročník: 2 ***Akademický rok:*** 2015/16

NÁZEV TÉMATU:

Trasování významných bodů ve videosekvenci nestacionární kamery

POKYNY PRO VYPRACOVÁNÍ:

Cílem práce je návrh a implementace algoritmů pro trasování významných bodů v obraze nestacionární kamery, konkrétně pro případ kdy je kamera držena a přesunována rukou.

1. Zpracujte literární rešerši týkající se sledování trajektorie významných bodů v dynamickém obraze.
2. Proveďte teoretický rozbor jevů, jenž mohou v obraze vznikat pohybem kamery a experimentálně Vaše předpoklady ověřte.
3. Navrhněte a popište alespoň tři různé algoritmy pro sledování trajektorie významných bodů, u nichž předpokládáte, že budou v zadané úloze použitelné.
4. Implementujte Vámi zvolené metody.
5. Navrhněte způsob hodnocení kvality trasovacích algoritmů a proveďte sérii experimentů pro její stanovení.
6. Zhodnoťte klady a zápory vytvořených metod.

DOPORUČENÁ LITERATURA:

[1] SONKA M., HLAVAC V., BOYLE R.: Image Processing, Analysis and Machine Vision. 3rd edition. Toronto : Thomson, 2008. 829 s. ISBN 978-0-495-08252-1.

[2] RUSS, J.C. The Image Processing Handbook. Boca Raton : CRC Press, 1995. 674 p. ISBN 0-8493-2516-1.

Termín zadání: 8.2.2016 ***Termín odevzdání:*** 16.5.2016

Vedoucí práce: Ing. Karel Horák, Ph.D.

Konzultant diplomové práce: Ing. Jan Klečka

doc. Ing. Václav Jirsík, CSc., *předseda oborové rady*

Abstrakt

Diplomová práce se zabývá problematikou trasování bodů získaných z videosekvence kamery, která je držena rukou. Práce je zaměřena na případ pohybující se kamery a statického pozadí a jevů, které jsou s tímto případem spojeny a mohou nastat. Je zde zkoumán pohyb kamery, který je dán jeho směrem a rychlostí. Jsou zde popisovány různé metody trasování významných bodů a objektů a také možnosti jejich použití. Cílem této práce je zvolení a následná implementace tří principiálně odlišných metod, vhodných k trasování významných bodů pro případ pohybující se kamery a jejich následné srovnání podle daných kritérií. Implementované metody byly detailně popsány spolu s nastavitelnými parametry metod. Byl ukázán vliv pohybového zkreslení na kvalitu snímku. Také byly pořízeny čtyři videosekvence za účelem testování metod dle zvolených kritérií. Na základě těchto kritérií jsou srovnány jednotlivé výhody i nedostatky zvolených algoritmů a také popsány možnosti uplatnění zvolených metod.

Klíčová slova

trasování, významný bod, KLT, vektor pohybu, pohybové zkreslení, popis scény, IPAN, RC, videosekvence.

Abstract

The thesis deals with the issue of tracking feature points earned from videosequences of hand helded camera. The work is focused on the case of moving camera and static background, and events that are associated with this case and can occur. There is studied the movement of the camera, which is given its direction and speed. The aim of this work is the election and the subsequent implementation of three fundamentally different methods suitable for tracking feature points in case of moving camera and their comparison according to set criteria. Implemented methods have been described in detail, along with adjustable parameters methods. It shows the influence of the motion blur on picture quality. Also were taken four movies for the purpose of testing methods according to selected criteria. On the basis of these criteria are compared individual advantages and disadvantages of selected algorithms and also discussed possibilities of selected methods.

Keywords

tracking, features, KLT , motion vector, motion blur, scene description, IPAN, RC, video sequence.

Bibliografická citace

STUDENÝ, P. Trasování významných bodů ve videosekvenci nestacionární kamery. Brno: Vysoké učení technické v Brně, Fakulta elektrotechniky a komunikačních technologií, 2016. 99 s.

Vedoucí semestrální práce Ing. Karel Horák, Ph.D..

Prohlášení

„Prohlašuji, že svou diplomovou práci na téma " Trasování významných bodů ve videosekvenci nestacionární kamery " jsem vypracoval samostatně pod vedením vedoucího diplomové práce a s použitím odborné literatury a dalších informačních zdrojů, které jsou všechny citovány v práci a uvedeny v seznamu literatury na konci práce.

Jako autor uvedené diplomové práce dále prohlašuji, že v souvislosti s vytvořením této diplomové práce jsem neporušil autorská práva třetích osob, zejména jsem nezasáhl nedovoleným způsobem do cizích autorských práv osobnostních a jsem si plně vědom následků porušení ustanovení § 11 a následujících autorského zákona č. 121/2000 Sb., včetně možných trestněprávních důsledků vyplývajících z ustanovení § 152 trestního zákona č. 140/1961 Sb.“

V Brně dne: **16.května 2016** ……

podpis autora

Poděkování

Rád bych poděkoval konzultantovi své diplomové Ing. Janu Klečkovi, za účinnou metodickou, pedagogickou a odbornou pomoc a další cenné rady při zpracování mé diplomové práce. Také za jeho ochotu a podnětné návrhy.

V Brně dne: **16.května 2016** ……

podpis autora

Obsah

Seznam obrázků

Seznam tabulek

1 ÚVOD

Počítačové vidění je vědní disciplínou, která se zabývá napodobením lidského zraku strojovými technikami. Technika pro snímání obrazu je nyní již na vysoké úrovni, ať už ve statické podobě (fotografie) nebo v dynamické (videosekvence). Problém zpracování obrazu je v tom, že neexistují jednoznačné postupy pro jednotlivé úlohy. Jinými slovy, kvůli rozmanitosti světa je obtížné definovat univerzální řešení. To platí i pro případ trasování bodů ve videosekvenci.

Úkolem diplomové práce je implementovat algoritmy, které jsou schopny ze dvou (či více) po sobě jdoucích snímků z videosekvence zjistit vzájemný posun jejich významných bodů a umožnit tak trasování. Předpokládá se, že pokud se ve scéně nenachází pohyblivé objekty, a pokud ano jsou ve stacionárním stavu. Jediný objekt, který se bude pohybovat, je kamera. Protože však bude kamera držená rukou, může docházet ke zkreslení obrazu, jehož příčiny a důsledky budou probrány.

Uspořádání této práce je takové, že po úvodu do problematiky tohoto tématu následuje seznámení a popis jednotlivých metod, které by mohly být použity k trasování a detekci významných bodů ve videosekvenci. A také možné požadavky, které klade řešení úlohy z této oblasti. Dále následuje implementace vybraných metod a následné porovnání těchto algoritmů z hlediska zvolených kritérií. Základem navržených algoritmů je, že jejich výstupem je množina trajektorií významných bodů. Při zpracování těchto výstupu je také možné rozpoznat, jakým směrem se kamera pohybuje. Implementované algoritmy by měly být co nejvíce odolné vůči případným zkreslením, které mohou při této situaci nastat (pohybové zkreslení).

2 ROZBOR PROBLEMATIKY

V této části práce bude proveden úvod do problematiky, která bude muset být řešena při zpracování a implementování trasovacích metod. Bude zde vysvětlen pojem videosekvence, popis možných zkreslení, které mohou nastat vlivem držení kamery rukou a také teoretické seznámení s možnými druhy trasování a různými situacemi, které mohou nastat mezi kamerou a pozadím. Další kapitola pak bude věnována podrobně některým druhům trasovacích algoritmů a detekcím významných bodů, které byly zvažovány pro použití v této práci.

2.1 Videosekvence

Na videosekvenci lze pohlížet jako na rychlý sled po sobě jdoucích statických snímků. Zpracování a přehrávání dynamických obrazových dat (videosekvence) bylo dlouhou dobu velkým problémem, na rozdíl od pořizování fotografií. Již od vzniku prvních zařízení pro přehrávání dynamických obrázků se využívalo nedokonalosti lidského zraku (setrvačnosti zrakového vjemu, rozlišovací schopnosti oka). Osvětlením oka dochází v sítnicí k podráždění nervových zakončení, které je v mozku vnímáno jako zrakový vjem. Zrakový vjem nevzniká ani nezaniká současně s jeho vyvoláním, ale o něco později (obnova podrážděných zakončení), tudíž při rychlých změnách jasu dopadajícího světla dochází ke sloučení s předchozím vjemem. Lidský mozek dokáže vnímat pouze střední hodnotu jasu, pokud je frekvence impulsů dopadající na oko nad určitou hranicí. Jinými slovy pro vznik vnímání souvislého přechodu mezi statickými obrazy stačí lidskému zraku vystavovat scénu po určitou dobu v rychlém sledu [1]. Filmová kinematografie zjistila, že pro vnímání plynulého pohybu je minimální hodnota snímkové frekvence 25 – 30 snímků za sekundu.

2.2 Účel a využití trasování ve videosekvenci

Trasování (sledování) významných bodů je jednou z úloh z oblasti počítačového vidění. Cílem trasování (z angl. tracking) je použití metod, kterými lze nalézt a sledovat pohyb bodů/objektů ve scéně snímané kamerou. Základem úspěšného trasování je nalézt a popsat algoritmus, který je schopen ze dvou po sobě jdoucích snímků z videosekvence zjistit vzájemný posun mezi jednotlivými body, respektive omezenými oblastmi, případně nalézt body v jednotlivých snímcích. Z takovýchto dat by poté bylo možné sjednotit body ve snímcích. Při zjištění vzájemného posuvu získáme vektor pohybu jednotlivých bodů ze snímků. Pro

detekci pohybu lze také použít optický tok. Všechny tyto metody a možnosti budou popsány později. Sledování objektů má mnoho aplikací v praxi.

Zjištění pohybu mezi snímky může být využito v různých oblastech, například:

1. Telematika (dohled nad silničním provozem, porušení pravidel)
2. Lékařství (laparoskopie, endoskopie)
3. Zabezpečovací technika (kamery pro detekci pohybu)
4. Automatické řízení strojů (řízení robotů, sond, umísťování DPS)
5. Videokomunikace (komprese dat)
6. Počítačová technika (rozšířená realita)
7. Meteorologie (sledování pohybu oblaků)
8. 3D rekonstrukce prostředí a další

Jako zajímavá ukázka využití trasování se jeví 3D rekonstrukce prostředí. Při 3D rekonstrukci jde o získání informací o trojdimenzionální scéně pomocí geometrického popisu vztahů mezi scénou a kamerami. Oblast, která se tímto geometrickým popisem zabývá, se nazývá vícepohledová geometrie "Multiple view geometry", pomocí níž lze určit například:

1. rekonstrukce prostředí z pohybu (SfM "Structure from motion")
2. rekonstrukce prostředí ze dvou snímků ("Stereo vision") .
3. 3D rekonstrukce scény nebo určení pozice kamery ("Simultaneous localization and mapping").

Příklad praktického využití je na **Obrázek č. 1**.

Obrázek č. 1 Structure from motion (ukázka) [2]

2.3 Významný bod v obraze

Tato sekce je věnována definici významných bodů, jejich detekcí a metodám hledání, které budou podrobněji popsány později. Nalezení, popis a následné porovnání mezi jednotlivými snímky je stěžejní část této práce, proto budou metody, které budou v práci implementovány podrobně popsány.

2.3.1 Významný bod – definice

Při zpracování obrazu hraje detekce významných bodů velkou roli. Významné body jsou používány při předzpracování obrazu, mohou být využity například pro rozpoznávání objektů, spojování obrazů, tvorbu panoramat, rekonstrukci tvarů objektů ve 3D a samozřejmě také při sledování trajektorie (trasování) pohybu různých bodů/objektů. Hlavním principem algoritmů pro detekci významných bodů je to, že se nezabývají obrazem jako celkem, ale pouze oblastmi, které nesou nějaké významné informace, díky nimž je tento bod a jeho okolí odlišné a může mít význam pro další zpracování obrazu. Takovýmto bodů se říká významné body (z anglické literatury image features). Význam těchto bodů je značné urychlení zpracovávání obrazových dat, jelikož je zpracování obrazových dat výpočetně náročné, je jednodušší místo porovnávání pixelu po pixelu porovnávat pouze tyto definované body a jejich blízké okolí.

Významný bod v obrazu je místo, které je co nejméně podobné svému blízkému okolí. Typickým představitelem významného bodu je roh, což je obrazový bod, kde se sbíhají dvě hrany dohromady. Detekce hran a rohů je zobrazena na Obrázek č. 2, dalšími představiteli jsou vrcholy, hranice a podobně. V těchto bodech má obrazová funkce velký gradient jasové funkce, proto je výhodné hledat významné body právě pomocí gradientu. Významný bod [3] tedy značí místo v obraze, které má:

• jasnou a matematicky dobře podloženou definici pro jejich nalezení

• jasně definovanou pozici v obrazovém prostoru

• okolí v obraze u významného bodu, které je bohaté na informace vhodné pro pozdější zpracování

• je stabilní z hlediska působení lokálních a globálních deformací v obrazové doméně, tak aby byl bod opět nalezen s vysokým stupněm opakovatelnosti. Jinými slovy je důležité, aby byl stejný významný bod nalezen ve stejném obraze i po působení fotometrických změn (např. při různé intenzitě světla nebo jasu) a také při působení transformací obrazu (změna měřítka, pootočení obrazu, zkosení, atd.).

Obrázek č. 2 Ukázka detekce některých významných bodů [4]

2.3.2 Detektory významných bodů

Požadavky kladené na významný bod jsou nyní definovány, další požadavky jsou kladeny také na detektory těchto významných bodů. Mezi tyto požadavky patří například související podmínka detekce všech správných bodů a vyhnutí se detekci falešných rohů (působením šumu, stínu atd.). Stabilita a robustnost detektoru s ohledem na šum, obrazové a fotometrické deformace vyplývá přímo z požadavků na detekci významných bodů. Tyto zásady pomáhají při následném určování korespondence mezi dvěma obrazy. Poslední požadavek na detektor je jeho rychlost a efektivita výpočtu, případně použití v real-time aplikacích. Jeho důraz je však závislý přímo na aplikacích jeho použití.

Detektory těchto významných bodů lze rozdělit na několik skupin [5] podle toho, co jednotlivé detektory v obraze hledají. Možné oblasti zájmů mohou být rohy (corners), hrany (edges), dále metody, které hledají regiony, vykazující ve svém okolí podobné vlastnosti jako např. barva, jas (ty regiony se označují v angl. literatuře jako blob) a také metody pracují na principu sledování změn zakřivení hran (ridges). Přehled detektorů významných bodů s rozdělením detekované oblasti zájmů je znázorněna v Tabulka 1. V této tabulce je zobrazen přehled nejčastěji používaných detektorů významných bodů, nejsou zde však uvedeny všechny, kvůli neustále se rozšiřujícím metodám a vylepšením. Jako příklad relativně nové, avšak velmi používané metody je metoda SURF prezentovaná poprvé v roce 2006. Jednotlivé metody v této části, které budou použity při implementaci, budou podrobněji rozepsány v kapitole Metody trasování bodů.

Feature detector	Edge	Corner	Blob	Ridge
Canny	X			
Sobel, Prewitt, Roberts	X			
Deriche	X			
Differential	X			
Kayyali	X			
Harris & Stephens / Plessey	X	X		
SUSAN	X	X		
Shi & Tomasi		X		
Level curve curvature		X		
FAST		X	X	
Laplacian of Gaussian		X	X	
Difference of Gaussian		X	X	
Determinant of Hessian		X	X	
MSER			X	
PCBR			X	
Grey-level blobs			X	
Hough transform				X
Structure tensor				X

Tabulka 1 Přehled detektorů významných bodů [5]

Problémy při detekci významných bodů u nestacionární kamery

Při hledání korespondencí mezi dvěma nalezenými významnými body však může dojít, vlivem působení rozmazání při pohybu kamery (motion blur), k nesprávné detekci bodu vlivem působení PSF funkce. Podrobněji bude tento problém popsán v kapitole 2.6.4. Výsledkem působení tohoto zkreslení je však možnost rozptylu bodu a jeho okolí ve směru pohybu kamery, což při následném hledání významného bodu v takto rozmazaném obrazu může vést k detekci jiného významného bodu. Tím bychom přišli o jeden, či více významných bodů, které by mohly korespondovat s body nalezenými v nezkresleném obraze. Jelikož by detektor mohl nalézt vlivem rozptylu zcela jiné body, než ty, jejichž „významnost“ by byla za normálních podmínek zřejmá.

2.4 Vektor pohybu, metody nalezení vektoru pohybu

V tomto úseku bude vysvětlen pojem vektor pohybu a jeho vztah ke sledování pohybu. Dále zde budou představeny metody, kterými se dá pohybový vektor nalézt. Některé z těchto metod budou také použity při trasování významných bodů.

Dále zde budou uvedeny obecné předpoklady pro zjednodušení detekce pohybu v obraze a tím i detekce významných bodů.

2.4.1 Vektor pohybu

Jedním z důležitých parametrů pro sledování pohybu významného bodu je pohybový vektor. Každý pohyb v obraze má totiž svoji velikost a směr, právě tyto parametry by měl pohybový vektor popisovat. Pohybový vektor se dá určit ze dvou po sobě jdoucích snímků z videosekvence, na které se buď pohybuje bod, nebo jako v našem případě kamera. Při použití detektoru významných bodů a nalezení korespondence mezi 2 body z rozdílných snímků lze tedy určit velikost a směr posuvu bodů z jednoho snímku na druhý. Pohybový vektor může popisovat vztah mezi celou scénou, mezi určitými objekty nebo jednotlivými pixely (významné body). Nevýhodou je to, že pohyb zachycený na kameře je pouze 2D, zatímco skutečně vykonaný pohyb se uskutečňuje ve 3D. Pro hledání pohybu v rovině (2D) by stačilo hledat pohyb v osách X, Y jako kombinace translace a rotace. V našem případě se však předpokládá 3D pohyb kamery a model pohybového vektoru bude hledán jako kombinace rotace a translace ve 3D a přiblížení s kamerou, nikoliv zoom kamery. Lze předpokládat, že jeden bod se mezi dvěma obrazy posune jen o určitou relativně malou vzdálenost.

2.4.2 Metody nalezení pohybového vektoru (analýzy pohybu)

V této sekci budou popsány metody, které jsou používány pro analýzu pohybu. Metody jsou rozděleny do 2 oblastí podle vztahu k poloze objektu, a to buď závislé, nebo nezávislé na objektu.

Nezávislá analýza pohybu je prováděna bez ohledu na polohu pohybujících se objektů/bodů. V této kategorii jsou nejčastěji používány metody optického toku, které lze díky své rozmanitosti a obsáhlosti považovat za vlastní kategorii.

Naopak u závislé analýzy je metoda závislá na detekci polohy objektů. U této metody se buď detekují významné body v obraze, nebo místa, kde nastal pohyb. A poté se hledá souhlas mezi takto nalezenými oblastmi či body, příkladem může být použití Harrisonova detektoru a následného hledání korespondencí.

V této části budou vypsány pouze metody používané pro analýzu pohybu, v další kapitole budou vybrané metody popsány podrobněji. Druhy analýzy jsou [6]:

1. **Nezávislé (na objektu)**
    - Bodově (pixelově) orientované
        1. Optický tok (Lucas-Kanade, Horn-Schunck)

2. fázová korelace, metody typu SoAD

- Blokově orientované (blokové porovnávání, sub-prostorové omezení)

2. **Závislé (na objektu)**
   - Korespondence bodů (detekce významných bodů + korespondence)
   - Rychlostní pole
   - Rozdílové metody

U objektově závislé analýzy je výhodné stanovit následující předpoklady [6]:

- Maximální rychlost – objekt/bod se může na následujícím obrazu zobrazit jen do omezeného okolí jeho předchozí polohy. Toto okolí je určeno kruhem o poloměru $v \cdot dt$ se středem v původní pozici bodu, kde v značí předpokládanou maximální rychlost pohybu objektu/bodu.
- Maximální zrychlení – velikost změny polohy mezi 2 po sobě jdoucími okamžiky je omezeno velikostí změny polohy předchozích okamžiků.
- Pohybová korespondence – topologické vlastnosti objektu se nemění (tuhá tělesa mají stabilní konfiguraci bodů, které se pohybují stejným směrem.

2.5 Popis scény

V tomto úseku budou popsány veškeré situace, které mohou při snímání pohybu v rukou držené kamery nastat. Dále budou popsány kombinace vzájemného postavení kamera-snímaná scéna a popsány problémy, které mohou nastat při obecné analýze pohybu (aperturní problém, problém projekce). Poté popíšeme efekty, které mohou nastat při situaci, kdy je kamera v pohybu a navíc držená rukou, což je jedním ze zadání této práce. Při této situaci může nastat zkreslení, které bude popsáno později.

2.5.1 Situace umístění kamery a snímané scény

Při analýze pohybu mohou nastat celkem 3 případy relativního pohybu soustavy kamera-objekty, které mají také vliv na použité metody detekce pohybu. Tyto situace jsou [6]:

- Kamera v klidu, objekty v pohybu
- Kamera v pohybu, objekty v klidu
- Kamera v pohybu, objekty v pohybu

Situací, kterou se bude tato diplomová práce zabývat, je případ, kdy se bude pohybovat pouze kamerou a předpoklad statického pozadí (objektů), neboli druhý případ. V některých případech jsou na pohybující se objekty kladeny určité požadavky nebo se zavádí jisté předpoklady o objektu a pozadí. Ideálními podmínkami pro analýzu pohybu je neměnné pozadí (barva, jas konstantní i při přesunu objektů) a hlediska objektů jsou nejvhodnější, když je v obraze jeden či více pohybujících se objektů, které jsou vzájemně oddělitelné a rozpoznatelné (konstantní jas a barva je u objektů předpokládána také). V praktickém použití však vznikají situace, které tyto předpoklady nesplňují. Vlivem:

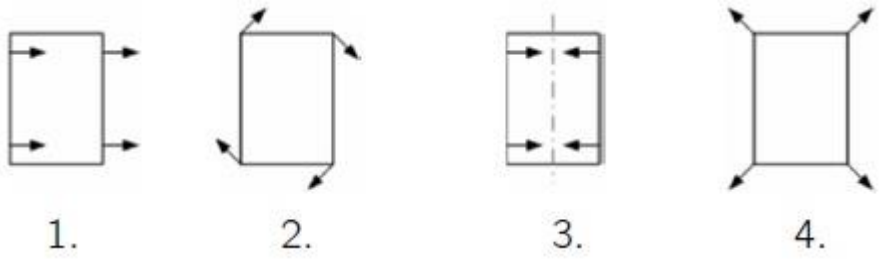
- Působení šumu v obraze.
- Počasí (náhlá změna jasu způsobená sluncem, oblaky, deštěm).
- Osvětlením (změna jasu vlivem rozsvícení/zhasnutí světel).
- Vlivem tzv. pohyblivých stínů, kdy vlivem změny úhlu osvětlení může dojít k prodloužení/zkrácení stínů, které se v obraze může registrovat jako pohyb, ačkoliv v reálném světe k pohybu nedošlo.
- Míjení objektů, sloučení objektů, ztráta objektu za statickou překážkou.
- Pohyb kamery (umístění na automobil, otřesy způsobené držením rukou).

2.5.2 Pohyb kamery (scény), pohyb objektu

Pohyb scény může být způsoben třemi situacemi. První situací je pohyb objektu uvnitř scény při statické kameře. Druhý případ, kterým se bude tato práce zabývat, je pohyb kamery při jinak statické scéně. Poslední a na řešení nejobtížnější možností je kombinace obou těchto pohybů. Výsledkem zjištění pohybu v obrazu je pohybový vektor, ve kterém je přiřazeno umístění a směr posunu každému bodu, u kterého se povedlo zjistit pohyb. Po získání tohoto vektoru pro všechny body se provede popis pohybu za pomoci vyhodnocení velikosti a směru posunu. Vyhodnotit jakýkoliv pohyb objektu v prostoru lze pomocí složení pohybu z kombinace základních 5 druhů pohybů [7].

Tyto základní pohyby jsou zobrazeny na **Obrázek č. 3** a jsou to:

1. Translace v konstantní vzdálenosti
2. Rotace v konstantní vzdálenosti
3. Rotace kolem osy kolmé na osu objektivu
4. Translace do dálky
5. Zkosení



Obrázek č. 3 Základní druhy pohybu [7]

Translační pohyb bude použit, pokud bude mít zjištěný pohyb bodů v obraze ve všech bodech přibližně stejný posun a velikost v jednom směru.

Pro přiblížení resp. oddálení kamery od scény se předpokládá zvětšení resp. zmenšení velikosti objektu (počtu bodů) a tím posuv bodů do všech směrů. V tomto případě se použije *translace do dálky*.

Pro *rotační pohyb* by musel mít posuv bodů v obraze stejnou velikost do všech směrů, ale s konstantním úhlem natočení. U rotačního pohybu se nemění poloha centra projekce. Popis *rotace kolem osy kolmé na osu objektivu* se používá např. při tvorbě panoramat, kdy dochází k otáčení kamery kolem své osy z 1 místa.

V této sekci bude ještě provedeno srovnání 2 již zmíněných druhů pohybu scény (pohyb objektu, kamery) a popis pohybových vektorů u obou těchto případů. Pro srovnání jednotlivých pohybů slouží Obrázek č. 4. Na snímcích je postupně zobrazen pohyb objektu a následně pohyb kamery.

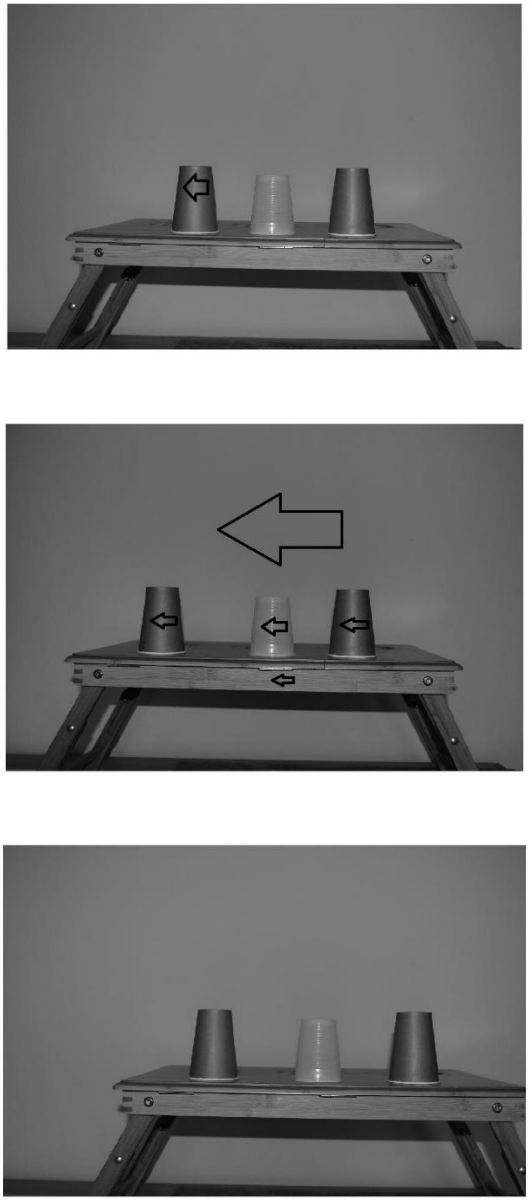
POHYB OBJEKTU

Na prvním a druhém snímku je zobrazen případ, kdy je pohybující se objekt zobrazen při statickém umístění kamery (kamera na stativu). Mezi těmito snímky je časová prodleva 1 sekunda, kdy došlo k posunutí kelímku. Pokud bychom pro tyto 2 snímky vytvořili snímek rozdílový, bylo by zřejmé, že k pohybu došlo pouze mezi novou a starou pozicí kelímku, přičemž pozadí zůstává statické. Pro toto pozadí bude pohybový vektor nulový, jelikož zde k pohybu nedošlo. Naopak na místech kde k pohybu došlo (kelímek), bude mít pohybový vektor svůj směr a velikost, a to pro každý bod kde k tomuto pohybu došlo.

POHYB KAMERY

Na druhém a třetím obrázku naopak objekty zůstávají statické, a dochází pouze k pohybu kamery do strany. Jelikož totiž v obrazu došlo k posunutí všech bodů na jednu stranu, zdá se, jako kdyby došlo k pohybu celé snímané scény. Všechny body těchto jednotlivých snímků ovšem budou posunuty současně a budou mít shodnou velikost i směr. Proto lze pohyb kamery popsat pouze jedním vektorem pro celý snímek. Směr tohoto vektoru bude opačný, než vektor pohybu kamery.

V tomto případě však sice dochází k pohybu kamery, ale ta je stále umístěna na stativu, přičemž se předpokládá její stálá vodorovná poloha. V našem případě (držení kamery rukou) však bude docházet, vlivem otřesů a chvění těla (viz. dále), k pohybu jak ve vertikální, tak horizontální ose. I když nemusí být chvění ve videu zřejmé, může se v rozdílovém snímku projevit jako detekování pohybu na hranách, kde dochází k velkému rozdílu jasových nebo barevných hodnot.



Obrázek č. 4 Srovnání pohybu objektu a pohybu kamery

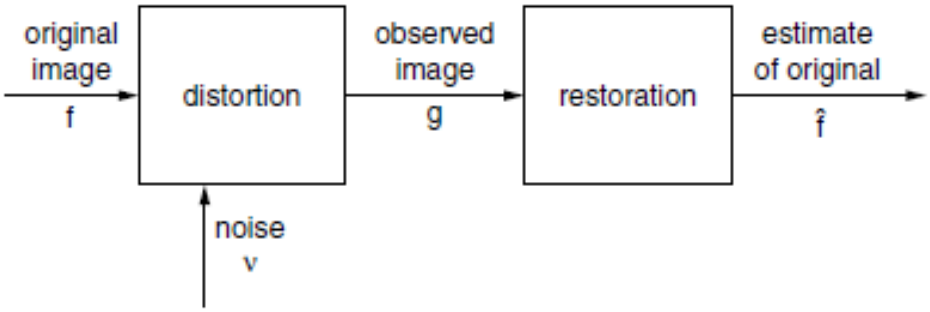
2.6 Zkreslení vznikající u nestacionární videosekvence

Tato kapitola je věnována zkreslení, které nastává při držení kamery rukou a které je označováno jako pohybové zkreslení, nebo pohybové rozmazání (z angl. motion blur). Bude vysvětleno, kdy k tomuto jevu dochází, příčiny jeho vzniku, parametry které tento jev ovlivňují a také metody, které pomáhají tento nežádoucí jev eliminovat. Odstraněním tohoto nežádoucího jevu se zabývá odvětví stabilizace obrazu. Protože se však v této práci budeme zabývat situací, kdy kameru držíme rukou, čímž odpadá možnost mechanické stabilizace. Jelikož se jedná o videosekvenci, odpadá nám možnost softwarové stabilizace obrazu, z důvodu velké výpočetní náročnosti zpracování. Proto mohou být použity metody restaurování obrazu, které dokáží obnovit původní obraz z obrazu rozmazaného pohybovým zkreslením. Ovšem i tyto metody jsou náročné.

2.6.1 Princip odstranění zkreslení z obrazu [8]

Před vysvětlením různých rušivých jevů, jako například třes kamery nebo právě pohybové rozmazání, bude obecně vysvětleno, jak se u takto ovlivněných záběrů dá působení těchto negativních parametrů odstranit. K tomuto odstranění slouží metody restaurování obrazu, jejichž princip bude stručně představen.

Nejprve se však seznámíme se systémem pro odstranění rušivých signálů (šum, pohybové rozmazání atd.) a obnovení původního obrazu (nebo jeho odhadu) před působením toho signálu známého jako restaurování obrazu. Cílem restaurace je odstranit zjištěné zkreslení z pozorovaného (observed) obrazu $\boldsymbol{g}$, a tak poskytovat nejlepší možný odhad $\hat{\boldsymbol{f}}$ originálního nezkresleného obrazu $\boldsymbol{f}$. Pozorovaný obraz může být zkreslen rozmazáním vlivem pohybu, geometrickými deformacemi, nelineárním přenosem kontrastu atd. a je obvykle dále degradován přídavným nebo jinak souvisejícím šumem $\boldsymbol{v}$. Identifikace vlastností zkreslení (tj. zkreslujícího systému, rušivého šumu) proto tvoří podstatnou část procesu restaurace. Po popsání zkreslení formálně matematickým modelem, s měřenými nebo odhadnutými parametry, se můžeme pokusit invertovat model a získat restaurovaný obraz (odhadnutý z originálu) jako výsledek aplikování inverzní procedury k pozorovanému (měřenému, získanému) obrazu. Schématická reprezentace zkreslení a restauračního procesu je vysvětlena na Obrázek č. 5.



Obrázek č. 5 Schématické znázornění procesu restaurace [8]

Metody identifikování modelů a parametrů zkreslení jsou specifické vůči individuálním druhům zkreslení.

V podstatě jsou v restaurování používány 2 přístupy. Koncepčně jednodušší z nich znamená formulování modelů zkreslení, které mohou být *přímo invertovány* (jako např. čisté lineární zkreslení); vyřešení systému rovnic nebo použití uzavřených formulí získaných inverzí pak přímo poskytuje odhad originálního obrazu. Pokud šum nemůže být zanedbán, exaktní inverze je nemožná, protože šum představuje neznámou stochastickou složku. V těchto případech je vyžadována přibližná inverze minimalizující vliv šumu; běžně používaný přístup je nejmenší chyba čtverců (LMS), která může vést také k uzavřenému vzorci (např. Wienerův typ filtrování).

Často jsou však reálné modely zkreslení tak složité (strukturálně a/nebo matematicky), že přímá inverze není proveditelná, nebo trpí příliš vysokými chybami. Proto se používá druhý případ, kdy se osvědčila postupná optimalizace zaměřená na extrémy z kriteriální funkce odvozené z modelu zkreslení. Výhody optimalizační teorie a iterativních algoritmů v kombinaci s teorií signálů tak poskytují silný nástroj, který často umožnuje obnovení užitečné informace dokonce i z velmi zkreslených obrazů.

2.6.2 Vznik třesu kamery [9]

Při pořizování záznamu na kameru/fotoaparát má na fotografa/kameramana vliv také okolní prostředí a také jako v našem případě jeho tělo. Tyto vlivy narušují podobu snímaného záběru, který se zkresluje. Těmito vlivy může být například foukající vítr, automobil při natáčení za jízdy, anebo jako v našem případě samotné držení kamery rukou, které způsobuje otřes kamery. V lidském těle totiž neustále dochází k přirozenému svalovému třesu (např. třes kosterního svalstva, odolávání působení gravitace), i když se zdá, že se člověk nebo jeho ruka nepohybuje a je ve stálé poloze. Druhý jev se projevuje při natáčení videozáznamu za chůze nebo běhu. Tělo člověka při chůzi, nebo běhu se totiž nepohybuje po přímé trajektorii,

ale dochází k pohybu v horizontálním i vertikálním směru zároveň a právě tyto pohyby způsobují možné rozmazání záběru. Tento třes těla je v reálných podmínkách u videozáznamu zřetelný při sledování krokovaného záběru. K odstranění tohoto třesu ze snímku slouží stabilizace obrazu.

2.6.3 Vznik, příčiny, parametry pohybového rozmazání [8,11]

Nyní když víme, za jakých situací dochází ke vzniku třesu kamery, musíme ještě zmínit, za jakých podmínek dochází k již zmiňovanému pohybovému rozmazání. Jak již název napovídá, pohybové rozmazání (motion blur dále jen MB) vzniká při pohybu. Tímto pohybem může být právě zmiňovaný třes kamery, ale také obyčejný pohyb kamerou při otočení nebo přemístění. Další možností je pohyb snímaného objektu/bodu. K MB dojde tehdy, pokud je některý z těchto pohybů vykonán během expozičního času (elektronický interval uzávěrky) při snímání kamery. Dá se tedy říct, že MB je funkcí expozičního času periody jednotlivých pixelů a pohybu/rychlosti pohybu (pohybového vektoru). Jelikož je expoziční doba parametr, který lze jednoduše nastavit a změnit, je dobré vědět, jak ovlivňuje toto zkreslení. Expoziční čas je totiž striktně spojen s MB a šumem. Přičemž platí, že zvolení této doby je vždy určitý kompromis mezi:

- Krátkou dobou expozice – menší možnost výskytu/redukce MB (možnost zanedbání), za cenu vyššího množství šumu.
- Dlouhou dobou expozice – redukuje šum, za cenu možnosti MB.

Proto je při focení/natáčení nutno zvážit, zda může dojít k MB (kvůli pohybu kamery/scény) a podle toho opatrně volit expoziční čas. Často se však nepodaří najít optimální nastavení. MB ovlivňuje velkou řadu algoritmů, které se zabývají pohybujícími sensory (registrace SFM, otřesy kamery, video analýza).

2.6.4 PSF, restaurování obrazu [8,10]

MB tedy závisí na expozičním čase a na pohybu, a to pomocí tzv. point spread function (PSF), neboli rozptylové funkci bodu. Tato funkce nám v podstatě říká, kam a jak dalece se může jednotlivý bod při rozmazání rozptýlit do svého okolí. Jak si nyní dokážeme, rozmazaný obraz vzniká konvolucí originálního (vstupního) obrazu s PSF. Rozmazání je způsobeno vlivem libovolného translačního paralelního pohybu celého obrazu $\boldsymbol{f(X, Y)}$ s ohledem na pole sensoru (nebo filmu) během doby expozice $\boldsymbol{t \in < 0, T >}$. Pokud se počátek obrazových souřadnic ***(X, Y)*** přesouvá do senzorových souřadnic ***(x, y)*** dle $x = X(t),\ y = Y(t)$. Získaný rozmazaný obraz je zjevně:

$$g(x,y) = \int_0^T f(x - X(t), y - Y(t))\,\mathrm{d}t\,. \qquad (1)$$

Spektrum takovéhoto obrazu je:

$$G(u,v) = \mathrm{FT}_{2D}\{g(x,y)\} = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty}\left(\int_0^T f(x - X(t), y - Y(t))\,\mathrm{d}t\right) e^{-j(ux+vy)}\mathrm{d}x\,\mathrm{d}y \qquad (2)$$

$$= \int_0^T e^{-j(uX(t)+vY(t))}\mathrm{d}t \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} f(\xi,\eta)e^{-j(u\xi+v\eta)}\mathrm{d}\xi\,\mathrm{d}\eta \quad = H(u,v)\mathrm{FT}_{2D}\{f(u,v)\}.$$

Využili jsme zde vzájemnou výměnu integračního pořadí a substituci **ξ = x-X(t), η=y-Y(t)**. Pak může být integrál rozdělen do 2 výrazů, závisejících jen na ξ, η a t. To znamená, že spektrum rozmazaného obrazu je tvořeno spektrem originálního obrazu s nějakou frekvenční přenosovou funkcí ***H (u, v***), která je plně určena pohybem během expozice. Proto je rozmazání opravdu konvoluční.

Aktuální PSF je často nepředvídatelná, jelikož se mění s ohledem na všech 6 stupňů volnosti pohybu kamery. Nicméně existují tyto techniky k odhadu PSF:

- analýzou samotného zachyceného obrazu
- pomocí výstupu akcelerometrů a gyroskopů
- dodatečným extrémně zašuměným obrazem s nízkou dobou expozice.

Přístupy závisí na jednotlivých (rozmazaných) obrazech a často vyžadují iterativní postupy s alternativním odhadem PSF, zatímco restaurovaný obraz je vždy získán dekonvolucí získaného obrazu a PSF (sledováním a následným odhadem). Restaurace provedená jakýmkoliv algoritmem závisí na několika souvisejících faktorech:

- Nejdůležitějším je čas expozice, který určuje vyrovnanost množství MB a šumu. Je důležité zdůraznit, že restaurovaný rozmazaný obraz není nikdy bez šumu, a dokonce male množství šumu může narušit efektivnost algoritmu restaurování.
- Další prvky, jako scéna nebo konkrétní trajektorie generující PSF, mohou významně ovlivnit provedení restaurace, ale jsou (narozdíl od doby expozice) těžší k ovládání. Zejména v řízených obrazových scénářích, kde vývoj PSF trajektorie může být spolu s expozičním časem statisticky studován nebo vyjádřen analyticky, může model restaurační chyby říct, zda zde existuje optimální expozice (tj. expoziční čas, který minimalizuje restaurační chybu).

3 METODY TRASOVÁNÍ BODŮ

Tato kapitola je věnována trasovacím metodám a algoritmům, které jsou založeny na detekci významných bodů. Budou zde také představeny jiné obecné metody, které jsou používány k trasování (bodů, objektů, vizuální, obličejové, videa) a také jen stručně popsány metody spojené spojeny s analýzou pohybu, pro kterou jsou používány podobné algoritmy, používané i pro trasování (korespondence bodů, metody optického toku). Následně budou vybrány 3 principiálně odlišné algoritmy, které budou dále podrobně popsány a poté v druhé části této práce implementovány do programu. Pro popis těchto trasovacích metod bude ještě blíže vysvětlen princip detektorů významných bodů, které budou použity za účelem trasování.

3.1 Metody analýzy pohybu

V této části budou stručně představeny přístupy sloužící k analýze pohybu. Některé metody, které k této analýze slouží, jsou totiž využívány také při trasování bodů, jsou to například metody optického toku (Lucas-Kanade využívaný pro algoritmus KLT), korespondence významných bodů. Kromě přístupů uvedených v 2.4.2, které byly děleny podle závislosti/ nezávislosti na objektu, se dá analýza pohybu rozdělit také podle metod, které jsou k této analýze používány. Mezi tyto metody patří [12]:

- Rozdílové metody (jednostranný, oboustranný, kumulovaný snímek)
- Optický tok (Lucas-Kanade, Horn-Schunck, Buxton-Buxton, Black-Jepson a další)
- Korespondence bodů (detekce významných bodů + přiřazení)
- Detekce specifických znaků pohybu (SLAM)
- Video tracking (kernel tracking – mean shift, contour tracking)
- Pohybové modely (Kalmánův filtr, Particle filter)

Metody, které nebudou použity pro trasování významných bodů, nebudou dále popisovány. Mezi tyto metody patří: rozdílové metody, detekce specifických znaků pohybu, video tracking a pohybové modely. Naopak budou představeny metody, které v této práci budou alespoň trochu zastoupeny tedy metody optického toku a korespondence významných bodů.

3.1.1 Optický tok [12,13]

Optický tok slouží pro určení pohybu a výsledkem jeho výpočtu je vektorové pole. Toto pole je složeno z dvojrozměrných vektorů rychlosti, které pro každý bod obrazu určují velikost a směr pohybu bodu, vzhledem k bodu na dalším snímku. Tyto vektory nemusí být určeny pro každý bod obrazu, ale je možné je určit jen pro některé významné body. Optický tok nám ovšem dává informace pouze o pohybu, který nastal v obraze, nikoliv o reálném pohybu ve snímané scéně. Korespondence s vektorovým polem pohybu (angl. motion field), které reprezentuje skutečný pohyb ve scéně, závisí na povaze snímané scény a způsobu snímání. Optický tok tedy zachycuje všechny změny v obraze za čas dt. Ovšem jedinou informací, ze které lze optický tok počítat, je intenzita, která se v čase mění. Lze tedy říct, že optický tok vyjadřuje posun intenzity v obraze. Tento posun však můžeme sledovat pouze tam, kde obraz obsahuje nějaké významné informace, neboli v místech, kde je gradient intenzity nenulový. Po výpočtu optického toku lze od sebe rozlišit základní pohyby popsané v kapitole 2.5.2.

Optický tok je metoda, která je založená na dvou předpokladech, kterými jsou:

• Konstantní intenzita - pixely ve scéně se skokově nemění, mezi jednotlivými snímky si uchovávají relativně stejný jas.

• Sousední body (náležící jednomu objektu/bodu) se pohybují stejným směrem.

Na *první podmínce* je založena celá myšlenka výpočtu optického toku a vyjadřuje tzv. předpoklad zachování intenzity. Tento předpoklad zavádí pro dynamický obraz jasovou funkci I(x, y, t), která určuje jas v místě (x, y), v čase t. Pro následující snímek tedy musí být splněna rovnice, která vychází z tohoto předpokladu tato rovnice je:

$$\mathrm{I}(x + dx, y + dy, t + dt) = \mathrm{I}(x, y, t) \qquad (3)$$

Cílem výpočtu optického toku je pro každý bod obrazu nalézt vektor pohybu, který je definován jako:

$$c = \left(\frac{dx}{dt}, \frac{dy}{dt}\right) = (u, v) \qquad (4)$$

Pokud by tento předpoklad nebyl dodržen, může dojít k chybné detekci pohybu.

Druhý předpoklad je využíván pro možnost rozpoznání jednotlivých bodů, objektů, které se mohou v obraze nacházet a které lze díky rozdílnému optickému toku dobře vzájemně rozlišit.

Optický tok lze počítat pomocí dvou základních přístupů.

První přístup spočívá ve výpočtu optického toku pro každý bod obrazu (z angl. Dense optical flow). Tento přístup je ovšem poměrně složitý a především časově náročný, zvlášť pokud se v obraze vyskytují předměty s homogenní plochou, u kterých dochází k výpočtu optického toku pouze na hranách objektu. Představitelem tohoto typu algoritmu je metoda Horn-Schunck.

Druhý přístup vychází z předpokladu, že bude optický tok počítán pouze pro body, které nesou důležitou informaci a dají se sledovat (významné body). Algoritmům tohoto typu se říká „sparse optical flow“, jejichž časová náročnost je podstatně kratší, než u algoritmů „dense optical flow“. Nejznámější představitel tohoto typu je metoda Lucas-Kanade, která bude dále popsána, jelikož je základem trasovacího algoritmu použitého v této práci (KLT algoritmus).

3.1.1.1 Metoda Lucas Kanade [14]

Metoda Lucas Kanade (dále jen LK) je široce využívaná diferenciální metoda optického toku typu „sparse optical flow“, která dokáže určit pohyb významných bodů v obraze. Tato metoda předpokládá, že tok je přibližně konstantní v blízkém okolí kolem uvažovaného významného bodu. Při výpočtu optického toku se využívá pouze malé okolí kolem významného bodu, tzv. integrační okno. Toto okno určuje velikost okolí kolem významného bodu, jelikož samotný bod nemá velkou informační hodnotu. Proto je nutné uvažovat i blízké okolí kolem významného bodu. Metoda je méně citlivá na obrazový šum, ale protože se jedná o čistě lokální metodu, nemůže poskytnout informace o toku uvnitř jednotného celistvého objektu (např. bílý papír), ale pouze na jeho hranách. Pro výpočet optického toku musí být opět splněny požadavky na optický tok (konstantní jas a malá vzdálenost mezi snímky, pohyb bodu podobný jako pohyb okolí). Druhá podmínka, která souvisí v diferenciálně malým pohybem mezi jednotlivými snímky může být zapsána formou rovnic. Vznikne nám tedy soustava rovnic, která bude mít takovou velikost, jaká je velikost zvoleného okna kolem významného bodu (např. pro okno 5x5 vznikne soustava 25 rovnic) ve tvaru:

$$I_x(q_1 \dots q_n) \cdot u + I_y(q_1 \dots q_n) \cdot v = -I_t(q_1 \dots q_n) \qquad (5)$$

Kde: $\boldsymbol{q_1, q_2 .. q_n}$ pixely uvnitř okna.

Tyto rovnice lze přepsat do matice ve formě:

$$\boldsymbol{A \cdot v = B}, \qquad \boldsymbol{kde\ A} = \begin{bmatrix} \boldsymbol{I_{x(q_1)}} & \boldsymbol{I_{y(q_1)}} \\ \cdot\cdot & \cdot\cdot \\ \boldsymbol{I_{x(q_n)}} & \boldsymbol{I_{y(q_n)}} \end{bmatrix}, \boldsymbol{v} = \begin{bmatrix} \boldsymbol{V_x} \\ \boldsymbol{V_y} \end{bmatrix}, \boldsymbol{B} = \begin{bmatrix} \boldsymbol{-I_{t(q_1)}} \\ \boldsymbol{-I_{t(q_2)}} \\ \cdots \\ \boldsymbol{-I_{t(q_n)}} \end{bmatrix} \qquad (6)$$

Hledaný vektor rychlosti $\boldsymbol{v}$ lze tedy vyjádřit z této rovnice ve tvaru:

$$v = (A^T \cdot A)^{-1} A^T \cdot B \qquad (7)$$

Toto řešení lze rozepsat do tvaru:

$$\begin{bmatrix} \boldsymbol{V_x} \\ \boldsymbol{V_y} \end{bmatrix} = \begin{bmatrix} \sum_i^n \boldsymbol{I_{x(q_i)}}^2 & \sum_i^n \boldsymbol{I_{x(q_i)}} \cdot \boldsymbol{I_{y(q_i)}} \\ \sum_i^n \boldsymbol{I_{x(q_i)}} \cdot \boldsymbol{I_{y(q_i)}} & \sum_i^n \boldsymbol{I_{y(q_i)}}^2 \end{bmatrix}^{-1} \cdot \begin{bmatrix} -\sum_i^n \boldsymbol{I_{x(q_i)}} \cdot \boldsymbol{I_{t(q_i)}} \\ -\sum_i^n \boldsymbol{I_{y(q_i)}} \cdot \boldsymbol{I_{t(q_i)}} \end{bmatrix} \qquad (8)$$

Pokud je $A^T \cdot A$ invertibilní, má tato rovnice řešení, a to za předpokladu, že je plného řádu, to znamená řádu dva. To platí, pokud se v ní vyskytují dvě velká vlastní čísla (λ_1, λ_2). Tato vlastní čísla jsou shodná s vlastními čísly, která bychom obdrželi při výpočtu významných bodů, pomocí Harrisova detektoru. Pokud jsou tedy obě tato vlastní čísla velká, jedná se o velký gradient jasu ve dvou na sebe kolmých směrech (roh). Body (rohy), které toto kritérium splňují, jsou nejvhodnějšími kandidáty k trasování a tento algoritmus je schopný s takovými body pracovat. Tato metoda je využívána u KLT trasování.

Nevýhoda tohoto přístupu spočívá v počítání optického toku pouze pro malé změny toku a to pouze do velikosti integračního okna. Rozšíření, které tento problém, řeší je použití pyramidové implementace. Ta umožňuje zachytit i daleko větší pohyby než je samotné integrační okno.

3.1.2 Analýza na základě korespondence významných bodu [12]

Základ této metody spočívá ve správné detekci významných bodů a následném nalezení odpovídajících si bodů v posloupnosti obrazů. Jinými slovy je potřeba v po sobě jdoucích snímcích nalézt významné body (které se vyskytují v obou obrazech) a na základě těchto bodů nalézt korespondenci mezi nimi. Jakmile máme významné body detekované, sestaví se dvojice potencionálně korespondujících bodů. Poté se iterační metodou počítá pro každou dvojici pravděpodobnost vzájemné korespondence, až do doby kdy, pro každý bod existuje značně vyšší pravděpodobnost korespondence, než s jinými významnými body z posloupnosti. Na začátku hledání korespondencí se mohou vyřadit ty body, které nesplňují základní předpoklady o pohybu tuhých těles (např. maximální rychlost těles). Tato metoda se ovšem používá pro snímky, kde je časové zpoždění mezi snímky velké. Proto tato metoda nebude použita, jelikož u videosekvence je časový interval snímání velmi krátký. Korespondenci tedy stačí hledat pouze v blízkém okolí předchozího snímku, nikoliv v celém obraze.

3.2 Metody trasování objektu (object motion tracking) [15]

Účelem trasování významných bodů je určit informace o pohybu kamery nebo pohybu objektu/bodu. Po detekci bodů můžeme zmapovat trajektorii, kterou objekt nebo některé významné body ze scény při pohybu vykonaly. Díky této trajektorii lze získat informace o předešlém pohybu. Díky těmto informacím poté můžeme následně předpovídat nebo odhadovat příští pohyb nebo také pozici objektu, bodu, nebo kamery. Aby bylo toto trasování podle významných bodů úspěšné, musíme tyto body nalézt v aktuálním a také předchozím obraze, abychom mohli určit pohybový vektor a tím i určit pohyb, který mezi snímky nastal. Podle souřadnic těchto bodů můžeme poté určit výslednou trajektorii a zmapovat vykonaný pohyb. Jednou z motivací pro použití trasovacích algoritmů je to, že samotná detekce objektu/bodu je často výpočetně náročná nebo náchylná k chybám. Trasování však může významně omezit oblast vyhledávání bodu/objektu, a tím urychlit výpočet. Výstup trasovacích algoritmů záleží na aplikaci a použité reprezentaci k popisu sledovaného bodu/objektu. V našem případě (trasování významných bodů) však bude hlavním výstupem trajektorie významných bodů. Trasování bodů/objektů má široké využití v oblastech jako například:

- Rozhraní člověk-stroj – trasování spojeno s analýzou pohybu člověka, rozpoznáním řeči, gest, výrazů tváře atd.
- Bezpečnostní odvětví – monitorování bank, obchodů, parkovišť a s nimi spojená detekce pohybu, případně tváře.
- Virtuální realita, počítačová grafika – počítačové hry, spojení pohybu člověka s virtuální postavou.
- Kódování obrazu – efektivnější skladování a přenos videa u telefonů.
- Lékařství – určení diagnózy a ohrožení zdraví.

Trasování bodů z videosekvence je jen jedna z mnoha technik určených pro trasování objektů. Patří do širší skupiny pasivních technik objektového trasování, které je založeno na měření signálů (světlo, zvuk). Další širokou technikou trasování objektů je aktivní trasování, které zahrnuje umístění senzorů na objekt, případně i kameru. Mezi aktivní trasování patří mechanické, vnitřní, ultrasonické, magnetické rádiové a hybridní trasování. Tyto techniky jsou vypsány jen pro získání přehledu o možnostech trasování a dále nebudou rozebírány.

Algoritmy trasování pohyblivého objektu lze zařadit do 2 základních skupin:

- 2D objektové trasování – získání informace o pohybu objektu/bodu, který se obecně pohybuje ve 3D prostoru z pořízeného 2D obrazu.
- 3D objektové trasování – odhad skutečného 3D pohybu objektu/bodu s využitím informací z obrazové sekvence (2D).

Oblast našeho zájmu bude pouze ve 2D objektovém trasování deformovatelných a tuhých objektů. Trasování 2D tuhých objektů se snaží určit pohyb objektů/bodů z obrazu, respektive posun pozice objektu/bodu za čas. Trasovací algoritmy lze také řadit podle charakterizujících prvků, které jednotlivé metody používají. Mohou to být například:

- Trasování určitého objektu – auto, část lidského těla atd.
- Počet pohledů (jedno-pohledová, více-pohledová, všesměrová technika)
- Stav kamery (pohybující se, stacionární)
- Prostředí trasování (vnitřní, venkovní)
- Počet trasovaných objektů (jeden, více objektů, skupina objektů) atd.

Při trasování v navazujících snímcích z videosekvence musíme použít určitý druh informace, který nám pomůže ke správnému sesouhlasení objektů/bodů mezi snímky. Musíme tedy ve snímcích nalézt a identifikovat vzájemné informace o obraze, které se vyskytují mezi snímky. Těmito informacemi mohou být například pozice, rychlost, barva, tvar a textura objektu/bodu. Na základě těchto informací lze poté rozdělit trasovací algoritmy, podle typu informací a nástrojů, které využívají při přiřazování mezi jednotlivými snímky. Rozdělení podle těchto nástrojů je:

- Metody založené na regionech (Region-based)
- Metody založené na konturách (Contour-based)
- Metody založené na šablonách (Template-based)
- Metody založené na významných bodech (Feature-based)

V následujících částech budou krátce popsány a vysvětleny principy prvních 3 metod. Poslední a pro naši práci nejdůležitější metoda založená na významných bodech bude vysvětlena a probrána podrobněji. Po probrání těchto metod budou popsány 3 principiálně odlišné metody, které budou v této práci implementovány a testovány.

3.2.1 Metody založené na regionech [15,17]

Tato trasovací metoda je vhodná k interpretaci a analýze pohybu získaného z videosekvence. Je založena na detekci regionů objektu, přičemž pojmem region se rozumí sada pixelů, které mají stejné (homogenní) vlastnosti. Tyto vlastnosti mohou být získány pomocí segmentace objektu pomocí charakteristických rysů objektu (barva, hrany), případně na pohybu získaném z videosekvence. Pojem region tedy značí oblast obrazu, která pokrývá objekt, jenž chceme sledovat.

Mezi nejvíce používané metody patří ty, které jsou založeny na informaci o barvě (Color-based). Mezi ty patří například: histogram barev a Pfinder, metody založené na zvolení barevného prostoru (RGB, HSV, CMY atd.). Barevná informace se ukázala jako velice efektivní, protože umožnuje rychlé zpracování, přičemž jsou výsledky dostatečně robustní pro provedení real-time trasování. Hlavním problémem u metod založených na segmentaci barvy a trasování je zajistit robustnost vůči změnám podmínek (odraz světla, stíny, tma), což je v otevřeném prostředí velice obtížné.

Další používanou metodou je rozdílová metoda založená na odčítání pozadí (Background subtraction). Tato metoda vychází z detekce pohybu. V prvním kroku je vytvořen model scény bez pohybujících se objektů tzv. pozadí a toto pozadí zůstává statické během videosekvence. Následující snímky jsou odečítány od pozadí, čímž dochází k detekci pouze pohybujících se objektů. Nicméně touto metodou je obtížné získat čisté informace vlivem prostředí scény a šumu kamery. Tento problém se často řeší použitím morfologických operací (dilatace, eroze). Uvedené metody patří mezi nejpoužívanější, existuje však spousta dalších metod.

3.2.2 Metody založené na konturách [15, 16]

Jinou cestou jak zprostředkovat trasování objektu je modelování objektu pomocí informací o rysech kontury objektu a jejich sledování ve videosekvenci. Výstup této metody je uzavřená křivka, která kopíruje objekt. Tato kontura sleduje pozici i tvar trasovaného objektu. Tato metoda je sice komplikovanější než trasování pomocí regionů, nicméně je mnohem robustnější. Využívá se např. v lékařství, bezpečnostním odvětví atd. Nejznámější a nejpoužívanější trasování je pomocí metody *aktivních kontur.* Jedná se o iterativní postup minimalizace energie kontury, která se deformuje vlivem vnitřních, vnějších a obrazových sil. Jinou možností je použití B-spline křivek, clustering a dalších.

3.2.3 Metody založené na šablonách [15]

Technika této metody je založena na principu přiřazování šablony k objektu (template matching) používaném při rozpoznávání objektu. Template matching může být definován jako proces hledání cílového obrazu (z videosekvence), jehož region nejvíce odpovídá regionu šablony. Při tomto procesu se využívá transformace geometrických souřadnic tak, aby byla minimalizována vzdálenost mezi šablonou a obrazem. V prvním kroku je nutné vytvořit model objektu, který chceme trasovat. V nejčastějším případě se tato metoda využívá u trasování lidské tváře. V tomto případě lze vytvořit šablonu online (např. sledování tváře uživatele po určitou dobu), nebo offline (vytvoření šablony z databáze obličejů). Po vytvoření šablony se poté provede template matching, při kterém se používají různé metriky vzdáleností, aby byla minimalizována odchylka od šablony např. Hammingova vzdálenost, SSD, normalizovaná korelace a další. Detekce objektu je však výpočetně velmi náročná a kvalita přiřazení závisí na detailech a stupni přesnosti šablony. Další časté využití je při trasování dopravních značek atd.

3.2.4 Metody založené na významných bodech (Feature-based)

Nyní se dostáváme k metodám, které tvoří základ této diplomové práce. Bude zde podrobněji vysvětlen princip trasovací metody založené na významných bodech a také uvedeny metody, které byly nalezeny na základě literární rešerše. Nakonec budou podrobně popsány 3 metody, které budou implementovány.

Trasování tohoto typu může být definováno jako pokus o získání parametrů pohybu z významných bodů získaných z videosekvence. Konkrétněji jde o získání parametrů spojených s rovinnou translací bodů, přičemž body v 2D prostoru se mohou posouvat, nebo rotovat s ohledem na hloubku. Jinými slovy je úkolem trasování tohoto typu získat pohybový vektor z významných bodů nalezených v aktuálním a předešlém snímku. Tak aby byla splněna rovnice [15]:

$$m_j = m_{j-1} + d_j \qquad (9)$$

Kde: m_j je pozice významného bodu aktuálního snímku

m_{j-1} je pozice významného bodu předešlého snímku

d_j je vypočtený pohybový vektor

Toto trasování může být implementováno velmi efektivně, i přes náchylnost bodů k posunu při trasování (tracking drift). Největším problémem trasování významných bodů je však správně určit a následně sledovat charakteristický bod obrazu tak, aby tento bod byl na všech snímcích stejný (stejný roh, hrana).

Obecně tedy při této metodě trasování musíme nalézt a zvolit pixely, které obsahují nějakou významnou informaci (rohy, hrany atd.) s ohledem na jejich okolí (jas, kontrast atd.), jelikož informace pouze z jednoho pixelu je nulová. Nejrozšířenější metodou pro určení významných bodů vhodných pro trasování je metoda s detekce minimálních hodnot vlastních čísel matice, označovaná jako Shi & Thomasi detektor, popsaná v „good features to track“ [18], která vychází z detekce rohů použitím Harrisova detektoru, ze kterého lze získat Harrisovu matici ve tvaru:

$$C(x,y) = \begin{bmatrix} \sum_W \boldsymbol{I_x}^2 & \sum_W \boldsymbol{I_x} \cdot \boldsymbol{I_y} \\ \sum_W \boldsymbol{I_x} \cdot \boldsymbol{I_y} & \sum_W \boldsymbol{I_y}^2 \end{bmatrix} \quad (10)$$

Kde: $\boldsymbol{I_x}, \boldsymbol{I_y}$ jsou gradienty obrazu pro uvažovaný bod ve směru x, y

$\boldsymbol{W}$ je okno o rozměru $n\ x\ n$, které má střed ve významném bodě.

Pokud provedeme výpočet vlastních čísel této matice, získáme rovnici pro detekci přítomnosti významného bodu ve tvaru:

$$H(\lambda_1, \lambda_2) = \lambda_1 \cdot \lambda_2 - \kappa \cdot (\lambda_1 + \lambda_2)^2 \quad (11)$$

V tomto případě mohou nastat celkem 3 případy velikosti vlastních čísel:

$(\lambda_1, \lambda_2) \approx 0$ pak pixel (x, y) z okna je konstantní (žádný významný bod).

(λ_1, λ_2) má jednu hodnotu blízkou nule, druhou vysokou, jde o hranu v obrazu.

(λ_1, λ_2) má obě hodnoty vlastních čísel vysoké, jedná se o roh v obrazu.

Později byl výpočet přítomnosti z rovnice (11) upraven (Shi & Tomasi) na tvar:

$$H(\lambda_1, \lambda_2) = \min((\lambda_1, \lambda_2)) \quad (12)$$

Jako bod vhodný pro trasování je uznán ten bod, který vyhovuje této podmínce a je vyšší, než určitá hodnota zvoleného prahu. Tato podmínka by měla zajistit, aby byl každý detekovaný významný bod z videosekvence více stabilní. Na základě této podmínky budou vyhledány významné body v této práci. Pro měření kvality významného bodu obrazu během trasování a jistotu, že je sledován stále stejný bod z videosekvence, se měří rozdílnost bodů, která odráží změnu vzhledu bodu mezi prvním a aktuálním snímkem.

Další metoda, která je spojená s procesem výběru vhodného bodu, je popsána v [19]. Je zde řešen problém s hledáním posunutí bodu o vzdálenost d z jednoho snímku na další a problém toho, že nelze trasovat pouze jediný pixel (šum, změna podmínek atd.), ale i jeho blízké okolí, které je definováno právě oknem pixelů. Tento problém je rozdělen na 2 směry. První směr se zabývá otázkou, jak poznat,

že trasujeme stále stejné okno, pokud se mění časem obsah obrazu. Tento problém je řešen pomocí sledování reziduí funkce, při kterém se sleduje, zda se příliš nemění vzhled pozorovaného okna. Druhý směr, kterým se práce ubírá, je spojený s měřením posunutí okna *d*. Otázka je, jak se při tomto posunutí kombinují rozdílné vektory rychlosti, abychom nakonec dostali jeden výsledný. Řešení spočívá v modelování změn okna při komplexnějších transformacích než translace například pomocí afinních map.

Další způsob [20] pro trasování významného bodu ze dvou po sobě jdoucích snímků je při použití křížové korelace, kde je prováděna korelace mezi nalezenými významnými body a jejich okny ve snímcích. Pro každý bod v novém snímku je ve čtvercovém hledán okolí nejlepší kandidát, který nejvíce odpovídá bodu z předchozího snímku. Nalezení nejvhodnějšího bodu je na základě maxima této korelace. Tato metoda je určena pro malé posuny ve snímcích (maximálně 10 pixelů). Přesnost metody záleží na přesnosti pixelů.

Jiný přístup k získání informací o typu objektu je uveden v [21]. Je zde použita kombinace adaptivní Houghovy transformace společně s metodou aktivních kontur k získání významných rysů z obrazu obličeje.

Tyto metody jsou stručně popsány v [15]. Nyní již existuje spousta nových metod, které z těchto uvedených vycházejí. Například existuje spousta variací a úprav metody KLT v její základní formě. Přehled základních trasovacích algoritmů a metod, které byly nalezeny v rámci rešerše literatury je vypsán v *Tabulka 2*, ze které jsou vybrány 3 principiálně odlišné metody. Tato tabulka obsahuje seznam metod získaných z [22, 23] a ostatních článků. Metody uvedené v této tabulce byly získány v rámci literární rešerše, existuje ovšem spousta dalších metod a algoritmů, které používají některé z uvedených metod a vylepšují je, případně jsou kombinacemi metod trasování významných bodů a např. trasování pomocí regionů. Existuje také spousta nových a odlišných algoritmů, které se již v rámci rešerše literatury nepodařilo dohledat, případně prostudovat a sepsat, jelikož spousta metod je kombinací více přístupů trasování. Zvolené algoritmy, které budou dále rozepsány a implementovány v práci jsou:

- KLT tracker
- IPAN algoritmus
- Feature tracking through subspace constraints

Název článků s metodami trasování	Autoři	Rok
1, An Iterative Image Registration Technique with an Application to Stereo Vision 2, Detection and Tracking of Point Features „KLT tracker“	1, Bruce D. Lucas and Takeo Kanade 2, Carlo Tomasi and Takeo Kanade	1981 1991
Tracking feature points: A new algorithm. „IPAN IP97“	Chetverikov and Verestóy	1998
Better feature tracking through subspace constraints	Bryan Poling, Gilad Lerman, etc	2014
Finding trajectories of feature points in a monocular image sequence.	Sethi and Jain	1987
Tracking feature points in time-varying images using an opportunistic selection approach.	Hwang	1989
Feature Point Correspondence in the Presence of Occlusion	Salari and Sethi	1990
Establishing motion correspondece	Rangarajan and Shah	1991
Tracking a Dynamic Set of Feature Points	Yi-Sheng Yao, Rama Chellappa, etc	1995
A feature tracking algortithm using neighborhood relaxation with multi-candidate pre-screening	Yen-Kuang Chen, Yun-Ting Lin, etc	1996
A Fast and Robust Point Tracking Algorithm	Veenman-Reinders and Baker	1998
Feature point tracking and trajectory analysis for video imaging in cell biology	I.F. Sbalzarini, P. Koumoutsakos	2005
A feature-based tracking algorithm for vehicles in intersections	Nicolas Saunier and Tarek Sayed	2006
Feature points tracking: robustness to specular highlights and lighting changes	M. Gouies, Christophe Collewet	2006
Stable 2D feature tracking for long video	Jong-Seung Park, Jong-Hyun Yoon, etc	2008
Robust feature tracking in image sequences using view geometric constraints	J. Lawver, A. Yilmaz	2013
Robust Self Localization by Edge Feature Tracking	Sezal Jain, Disha Prakash, Venkatesh K Subramanian	2013
Eigenspace-based Tracking for Feature Points	Chen PENG, Qian CHEN, etc	2014

Tabulka 2 Přehled trasovacích metod získaných z rešerše literatury

3.3 metoda Kanade-Lucas-Tomasi (KLT)

Trasovací algoritmus KLT patří mezi nejpoužívanější algoritmy, je rozšířený a jeho implementaci v dnešní době obsahuje každý programový jazyk s toolboxy na počítačové vidění (C, Matlab atd.), přičemž se jednotlivé přístupy k této metodě mohou lišit vlivem různých vylepšení této metody (pyramidová implementace, kompoziční, aditivní algoritmus a další jak uvádí [24]). V této kapitole bude detailně popsán základní algoritmus této metody.

Základ toho algoritmu tvoří dřívější práce Lucase a Kanadeho uvedená v publikaci [25], která řeší otázku, jak trasovat body z jednotlivých po sobě jdoucích snímcích a navrhli metodu pro trasování oblastí (patch) v obrazu. Později byl algoritmus plně rozvinut v článku [19], přičemž v této publikaci byla navrhnuta metoda, jak nejlépe zvolit významný bod (feature). Výsledný algoritmus je jasně vysvětlen v článku [18] s upřesněním algoritmu pro nejvhodnější body k trasování. Algoritmus popsaný v [19] je považován za základní (originální) KLT algoritmus, který bude v této kapitole popsán. Existuje však mnoho vylepšení toho algoritmu, jak bylo uvedeno výše.

Algoritmus je založen na několika krocích. Prvním je nalezení bodů vhodných pro trasování, jenž jsou nalezeny pomocí minima vlastních hodnot matice, stručně popsané v 3.2.4. Dalším krokem je trasování bodů, které splňují toto kritérium pomocí iterační metody Newton-Raphson, která minimalizuje rozdíl mezi dvěma okny. Jinými slovy tento algoritmus měří podobnost oken, pomocí kvadrátu L_2 normy a to tak, že minimalizuje chybu mezi obrazovým vzorem a vstupním obrazem pomocí iterativní metody.

Cílem tohoto algoritmu je přiřadit šablonu obrazu (získanou v předešlém obraze) ke vstupnímu (aktuálnímu) obrazu pomocí okénka. K dispozici tedy máme:

- $T(x) -$ šablona obrazu získána vybráním okénka, které překrývá nalezený významný bod (případně objekt) a to v čase t.
- $I(x) -$ okénko z následujícího obrazu, které je ve stejné pozici jako okénko šablony, ovšem je získáno z obrazu v čase $t + 1$. V tomto čase se již sledovaný bod posunul do jiné pozice.
- $x = [x, y]^T -$ sloupcový vektor souřadnic v obraze.
- $W(x; p) = [x + p_x; y + p_y]^T -$ množina dovolených translací okénka.
- $p_x, p_y -$ posunutí bodu ve směru x, y o hodnotu p (parameter pohybu).

Nyní se algoritmus snaží posouvat okénkem tak, aby došlo k minimalizaci rozdílu mezi touto dvojicí okének, přičemž posouvání okénka je vymezeno maticí $W(x;p)$, která určuje veškeré možné posunutí. Jelikož je v této situaci k dispozici více rovnic než proměnných, hledá se podobnost okének $T(x)$ a $I(x)$ pomocí metody nejmenších čtverců. Nejlepší pozice je určena pomocí gradientní iterativní metody Newton-Raphson, kdy se okénkem posunuje v každé iteraci do doby, než je minimalizována kvadratická odchylka mezi $T(x)$ a $I(x)$. Tuto situaci popisuje rovnice (13):

$$\sum_x [\, \boldsymbol{I}\big(\boldsymbol{W}(x;p)\big) - \boldsymbol{T}(x)\,]^2 \qquad (13)$$

K minimalizaci dojde při posunutí okénka o optimální velikost Δp, viz(14):

$$\sum_x [\, \boldsymbol{I}\big(\boldsymbol{W}(x;p+\Delta p)\big) - \boldsymbol{T}(x)\,]^2 \qquad (14)$$

Tento výraz je nelineární, i když jsou dovolené translace lineární, hodnota pixelů obecně lineární není, protože v podstatě nesouvisí s x. Tuto rovnici tedy použitím Taylorova rozvoje prvního řádu linearizujeme vzhledem k Δp, čímž dostaneme:

$$\sum_x [\, \boldsymbol{I}\big(\boldsymbol{W}(x;p))\big) + \nabla I \frac{\partial W}{\partial p}\Delta p - \boldsymbol{T}(x)\,]^2 \qquad (15)$$

Kde: $\nabla I = [\frac{\partial I}{\partial x}, \frac{\partial I}{\partial y}]$ - je gradient obrazu ve směrech x, y vypočtený z $W(x;p)$

$\frac{\partial W}{\partial p} = \frac{\partial [x+p_x; y+p_y]^T}{\partial [\,p_x; p_y]^T} = \begin{bmatrix}1 & 0\\ 0 & 1\end{bmatrix}$ je Jacobiho matice translací ve směru x, y.

Pro nalezení minima výrazu derivujeme rovnici (15) podle Δp, tím získáme:

$$2\sum_x [\nabla I \frac{\partial W}{\partial p}]^T [\, \boldsymbol{I}\big(\boldsymbol{W}(x;p))\big) + \nabla I \frac{\partial W}{\partial p}\Delta p - \boldsymbol{T}(x)] \qquad (16)$$

Vyjádřením posunutí Δp z této rovnice dostaneme tvar:

$$\Delta p = H^{-1}\sum_x [\nabla I \frac{\partial W}{\partial p}]^T [\, \boldsymbol{T}(x) - \boldsymbol{I}\big(\boldsymbol{W}(x;p))\big)] \qquad (17)$$

Kde: $H = \sum_x [\nabla I \frac{\partial W}{\partial p}]^T \left[\nabla I \frac{\partial W}{\partial p}\right] = \sum_x \begin{bmatrix}1 & 0\\ 0 & 1\end{bmatrix}\begin{bmatrix}I_x\\ I_y\end{bmatrix}[I_x \quad I_y]\begin{bmatrix}1 & 0\\ 0 & 1\end{bmatrix} = \begin{bmatrix}\sum_x \boldsymbol{I_x}^2 & \sum_x \boldsymbol{I_x}\cdot\boldsymbol{I_y}\\ \sum_x \boldsymbol{I_x}\cdot\boldsymbol{I_y} & \sum_x \boldsymbol{I_y}^2\end{bmatrix}$
– je aproximace Hessovy matice, která je stejná jako Harrisova matice. Na tomto základě je vytvořena podmínka pro výběr vhodných bodů k trasování, jak již bylo uvedeno v kapitole 3.2.4.

Výpočet posunutí z rovnice (17) je počítáno v každé iteraci. Jakmile je posunutí spočítáno, můžeme aktualizovat parametr celkového posunutí okna ve smyslu:

$$p \rightarrow p + \Delta p \qquad (18)$$

Ukončit aktualizace parametru můžeme například podmínkou konvergence:

$$\|\Delta p\| \leq \epsilon \qquad (19)$$

Pro správnou funkci tohoto algoritmu je nutné pracovat v subpixelové přesnosti. V opačném případě většinou algoritmus selhává. Pokud algoritmus v daném počtu kroků nekonverguje, je lepší tento bod z dalšího sledování vyřadit, protože s velkou pravděpodobností jsme nalezli jiný než skutečný posun. Celý algoritmus lze shrnout do několika následujících kroků:

Najít vhodné body $\boldsymbol{x}$ k trasování pomocí Harrisovy matice tak, aby vyhovovaly rovnici $H(\lambda_1, \lambda_2) = \min((\lambda_1, \lambda_2))$. Poté zvolit vzorek $T(\mathbf{x})$ v okolí významného bodu $\mathbf{x}$ v minulém obrázku, nastavit posunutí $\mathbf{p} = [0; 0]$ a iterovat:

1. Vyber vzorek $I(\mathbf{W}(\mathbf{x}; \mathbf{p}))$ z nového obrázku v okolí harrisova bodu $\mathbf{x}$ s uvažováním aktuálního posunutí $\boldsymbol{p}$
2. Vypočti obrazovou chybu $E = I(\mathbf{W}(\mathbf{x}; \mathbf{p})) - T(\mathbf{x})$
3. Vypočti gradienty ∇I s translací $\mathbf{W}(\mathbf{x}; \mathbf{p})$.
4. Vypočti Jakobián $\frac{\partial W}{\partial p}$ v bodě (x; p)
5. Vypočti matici $\left[\nabla I \frac{\partial W}{\partial p}\right]$
6. Aproximuj Hessovu matici H skalárním součinem $H = \sum_{\mathbf{x}} [\nabla I]^T [\nabla I]$.
7. Odhadni translaci $\Delta \mathbf{p} = H^{-1} \sum_{\mathbf{x}} [\nabla I]^T [T(\mathbf{x}) - I(\mathbf{W}(\mathbf{x}; \mathbf{p}))]$
8. Aktualizuj parametry použitím $\mathbf{p} \leftarrow \mathbf{p} + \Delta \mathbf{p}$
9. Testuj konvergenci (iteruj), dokud není splněno $||\Delta \mathbf{p}|| \leq \epsilon$.

Poté nastavit novou pozici Harrisova bodu $\mathbf{x} \leftarrow \mathbf{x} + \mathbf{p}$ a proces opakovat pro další obraz. Tento algoritmus je shrnut v [24] spolu s dalšími úpravami.

Implementace tohoto algoritmu bude využita v této práci, pro srovnání s dalšími vybranými metodami. Implementace bude provedena s použitím optimalizované funkce pro výpočet toho algoritmu v programovacím jazyku Matlab, případně C.

3.4 Trasovací algoritmus IPAN IP97 [23, 26]

Tento algoritmus je založen na trasování významných bodů (feature point- based) v dlouhých videosekvencích. Je určen pro dynamické scény s uvažováním více nezávisle na sobě se pohybujících předmětech, ve kterých mohou významné body vstoupit, chvilkově zmizet a opustit zorné pole. Některé existující algoritmy pro trasování významných bodů (Sethi & Jain, Hwang, Rang. & Shah, Salari & Sethi) mají omezené možnosti pro zvládání neúplných trajektorií, zejména pokud je počet bodů a jejich rychlost pohybu vysoká a má nejednoznačnou trajektorii. Tento algoritmus byl navržen tak, aby tyto nejednoznačnosti účinně řešil. Základ tohoto algoritmu je uveden v [26] a také stručně popsán v [23].

Základní myšlenka IPAN algoritmu spočívá (podobně jako u většiny trasovacích algoritmů) v předpokladu, že trasovaný bod může patřit pouze k jediné trajektorii. Další důležitý předpoklad, který tento algoritmus využívá, je použití odhadu pro maximální rychlost v_{max} pohybu objektu/bodu. Doba trvání zastínění objektu/bodu je omezena na 2 snímky, toto omezení se ovšem může zvýšit. Algoritmus je založen na myšlence konkurujících si trajektoriích. Korespondence s předchozím a následujícím snímkem jsou stanoveny jako výsledek procesu testování konkurenčních hypotéz, které jsou podobné hypotéze použité v originálním procesu přiřazování 2 snímků [27]. Základním rozdílem je, že tři po sobě jdoucí snímky jsou nyní používány k získání dvou vektorů posunu potřebných k vytvoření hodnocené funkce „cost function“ založené na hladkosti.

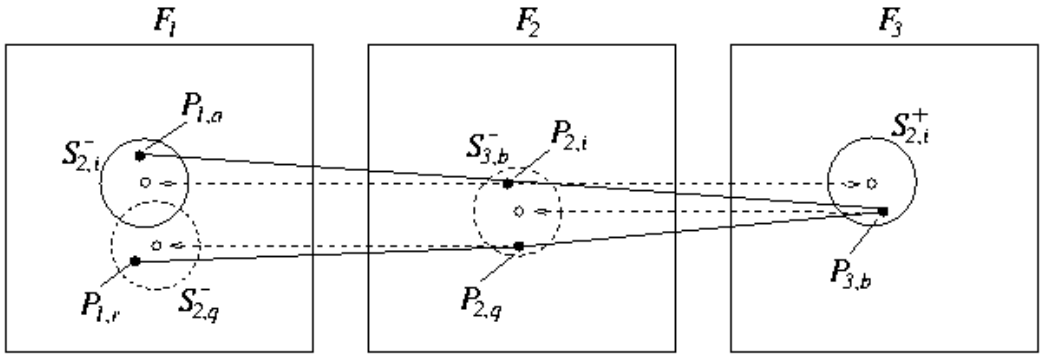
Přiřazovací algoritmus má následující tři kroky:

- Inicializaci
- Zpracování následných snímků
- Post-processing.

Proces inicializace pracuje s prvními třemi snímky a zahajuje proces trasování. Body ve snímku F2 jsou spojeny s odpovídajícími si body v F1 a F3. Začátek trasování startuje snímkem F3, kdy se začíná používat jiný postup přiřazování (matching). Jakmile jsou přiřazeny všechny snímky, je použit post-processing postup, aby se přehodnotili body, které dočasně zmizeli a později se znovu objevily. Tento postup se pokouší spojit korespondující koncové body z přerušených trajektorií. Jednotlivé kroky budou dále popsány.

3.4.1 Inicializace

Prvním krokem algoritmu je inicializace, která je znázorněna na Obrázek č. 6. Plné kruhy jsou pohyblivé body, prázdné kruhy jejich předpokládané umístění v sousedních snímcích. Plné čáry ukazují konkurenční trajektorie a přerušované čáry znázorňují vytvoření vyhledávacích oblastí.



Obrázek č. 6 Proces inicializace [26]

Vezměme libovolný bod $P_{2,i}$ ve snímku F_2, a pokusme se najít korespondující body v F_1 a F_3. Při použití předpokladu o omezení maximální rychlosti, se projeví možné umístění bodu $P_{2,i}$ na snímcích F_1 a F_3 a tím se určí oblast vyhledávání bodu $P_{2,i}$ ve snímcích F_1 a F_3, jako ty regiony, které mohou obsahovat kandidáty korespondujících bodů k $P_{2,i}$. Označme $S^-_{k,i}$ a $S^+_{k,i}$ jako vyhledávací oblasti $P_{k,i}$ v F_{k-1} a F_{k+1}. Poloměr těchto kruhových oblastí je definován jako maximální rychlost v_{max}.

Pak se vezmou v úvahu všechny možné trojice bodů ($P_{1,n}$, $P_{2,i}$, $P_{3,m}$), kde $P_{1,n} \in S^-_{2,i}$, $P_{3,m} \in S^+_{2,i}$, které mohou obsahovat bod $P_{2,i}$. Poté se najde trojice ($P_{1,a}$, $P_{2,i}$, $P_{3,b}$), která pro $k = 2$ minimalizuje *cost* funkci δ ($P_{k-1,n}$, $P_{k,i}$, $P_{k+1,m}$).

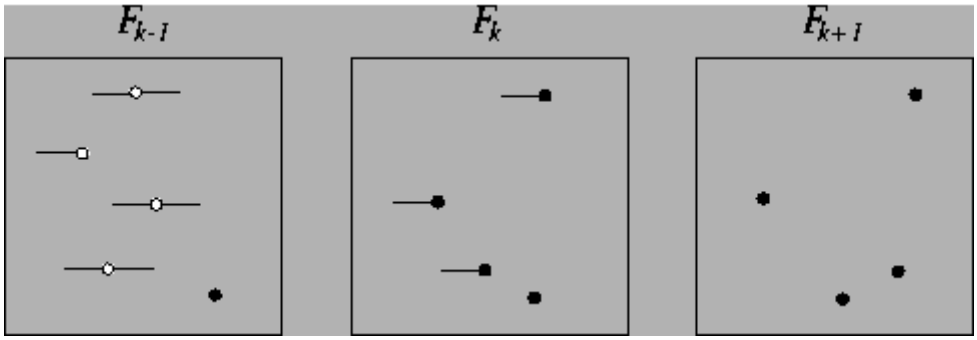
Hodnotící *cost* funkce vyhodnocuje lokální odchylku od hladkosti křivky a penalizuje změny v obou směrech a velikosti vektoru rychlosti. Hodnota funkce je normalizována tak, aby byla v rozmezí [0,1], přičemž 0 znamená bez změny.

Zvolíme trojici ($P_{1,a}$, $P_{2,i}$, $P_{3,b}$) jako počáteční hypotézu a zařadíme zbývající trojice podle hodnot jejich hodnotící funkce (jsou uznány jen trojice s $\delta < \delta_{max}$). Otestujeme počáteční hypotézu prohledáním $S^-_{3,b}$ (hledací oblastí $P_{3,b}$ z F_2). Nechť je $P_{2,q}$ bod v $S^-_{3,b}$. Pokud $S^-_{2,q}$ má bod $P_{1,r}$ takový, že $\delta(P_{1,r}, P_{2,q}, P_{3,b}) < \delta(P_{1,a}, P_{2,i}, P_{3,b})$, je počáteční hypotéza zamítnuta a je testována druhá nejlépe hodnocená hypotéza. V opačném případě pokračuje testování počáteční hypotézy s kontrolou bodu $P_{1,a}$ stejným způsobem. Pokud je i tato kontrola úspěšná, je ($P_{1,a}$, $P_{2,i}$, $P_{3,b}$) výstupem počáteční části trajektorie $P_{1,a}$. Stanovené korespondence se při dalším zpracování nemění.

Pokud jsou všechny hypotézy pro $P_{2,i}$ zamítnuty, není tento bod přiřazen k F_1 ani F_3. Je však stále možné, že $P_{2,i}$ bude přiřazen k F_3 při pozdějším zpracování. V takovém případě se $P_{2,i}$ objeví a otevírá trajektorii. Když jsou zpracovány všechny body snímku F_2, mohou některé body v sousedních snímcích zůstat nepřiřazeny. Bod v F_1, který není přiřazen žádnému bodu v F_2, mizí, to znamená, že buď opustil zorné pole, nebo dočasně zmizel ze snímku (například kvůli zastínění). Nepřiřazený bod v F_3 může později otevřít trajektorii. Ve výše uvedeném popisu procesu testování hypotéz může být původní hypotéza okamžitě zamítnuta, pokud bude nalezena „silnější" trojice ($P_{1,r}$, $P_{2,q}$, $P_{3,b}$). Nicméně trojice ($P_{1,r}$, $P_{2,q}$, $P_{3,b}$) může být stále předběhnuta jinou trojicí. Pokud ověřovací proces při testování ($P_{1,r}$, $P_{2,q}$, $P_{3,b}$) bude obdobný, může se přepnout zpět k F_2 a F_3. Původní hypotézu lze případně obnovit. Toto hlubší ověřování předpokládá delší videosekvence s podobnými událostmi, jejichž pravděpodobnost klesá s *hloubkou ověření* N_{ver}. V této sekci je použito $N_{ver} = 1$.

3.4.2 Zpracování následných snímků

Tento algoritmus se používá po inicializaci, a pracuje s následujícími snímky po prvních třech snímcích (F_1, F_2, F_3). Bod v každém snímku může mít nejvýše 2 linky (jednu dopředou, jednu zpětnou). Linka značí, že bod je propojen se sousedním snímkem. Každá linka má přiřazený vektor posunutí. Postup přiřazování pro $k > 2$ také pracuje se 3 po sobě jdoucími snímky. Pracuje s aktuálním snímkem F_k, $k > 2$ a uvažuje, že předchozí snímek F_{k-1} byl zpracován. V F_{k-1} se objevují body s nulovou (žádnou) linkou (Z-body), k těmto bodům může být stanovena potencionální korespondence v F_k. Všechny body s jednou linkou jsou spojeny s F_{k-2}, to znamená, že jsou zpětně propojené body (zkráceně B-body). Tyto body jsou označeny jako zmizelé a jsou zvažovány pouze v kroku post-processingu. Většina bodů má obvykle dvě linky, které indikují pokračující trajektorii. V F_k může mít bod buď zpětný, nebo žádný odkaz. V F_{k+1} jsou všechny body volné (jak ukazuje Obrázek č. 7).



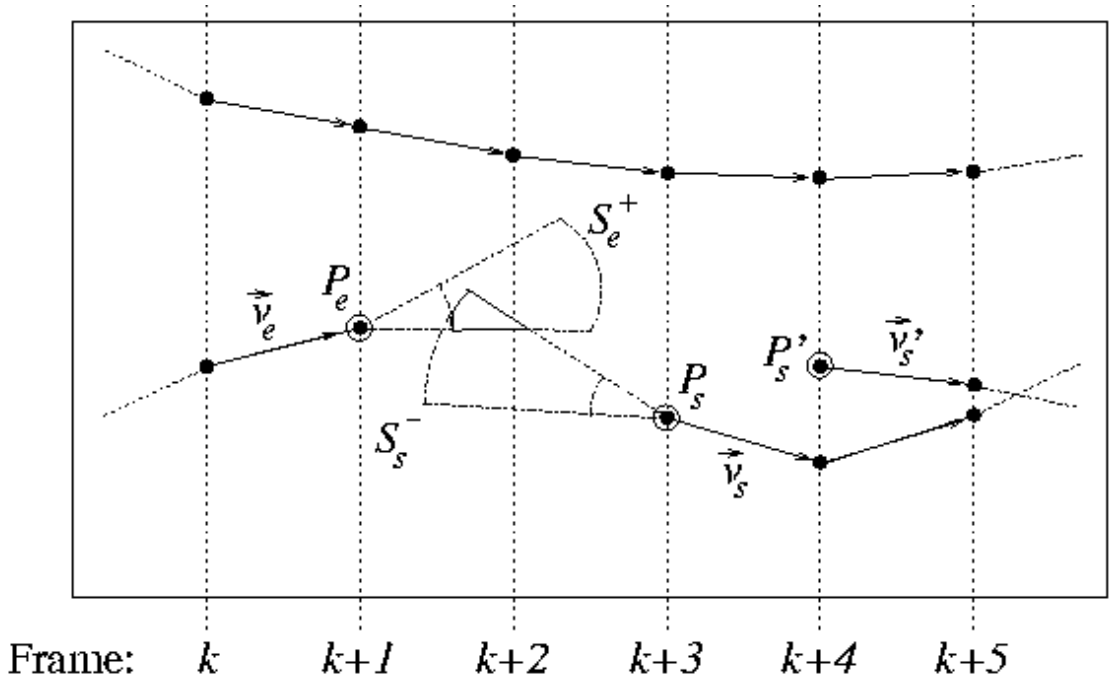
Obrázek č. 7 Proces zpracování snímků [26]

Významné body v F_k jsou zpracovány podobně jako v kroku inicializace popsané výše. Jediný rozdíl je ten, že pokud jsou k dispozici, jsou již použity stanovené korespondence. Tyto korespondence nejsou upravovány. V důsledku toho, během testování hypotéz B-bodů z F_k snímku, podporuje jejich předchozí posunutí, zatímco Z-body jsou promítány zpětně na F_{k-1} snímek, aby nalezli jejich kandidáty k posunu. V F_{k-1} jsou zvažovány pouze Z-body. Tyto body mohou být spojeny v F_k a stát se dopřednými linkami (F-body).

Tento postup poskytuje přirozený způsob jak zvládnout objevující se a mizící body včetně pohybu přes hranice obrazu. Pohybující se body navazují své linky v procesu soutěžení, který se vyvíjí s rostoucí trajektorií. Když je zpracován poslední snímek, body dvojitými linkami vytváří pokračující trajektorii. B-body jsou mizející, F-body se objevují.

3.4.3 Post-processing ztracených trajektorií

Tento postup se snaží propojit trajektorie, které byly ztraceny (rozbité). Na Obrázek č. 8 je ukázka rozbité trajektorie spolu s 2 možnostmi (pokračující, nebo oddělená) pokračující trajektorie. Zvažme B-bod P_e s příchozí rychlostí $\overrightarrow{v_e}$ a F-bod P_s s odchozí rychlostí $\overrightarrow{v_s}$. Kandidát zastíněného bodu je hledán v průsečíku dvou vyhledávacích oblastí S_e^+ a S_e^-.



Obrázek č. 8 Post-processing [26]

Oblasti hledání jsou v podstatě definovány pomocí hodnotící funkce (limit hodnoty a limit rychlosti). Kromě toho tento postup využívá fakt, že současné zastínění a prudké otočení jsou vzácné. Aby bylo zajištěno pokračování směru ztracené trajektorie, průnik $S_e^+ \cap S_e^-$ je omezen více, než je předepsáno limitem hodnoty. Pokud je průnik $S_e^+ \cap S_e^-$ prázdný, trajektorie zůstává ztracena. V opačném případě je nalezen bod, který minimalizuje hodnotu pro interpolovanou trajektorii. To se provádí důkladným prohledáváním průniku $S_e^+ \cap S_e^-$ s vhodným prostorovým rozlišením. Pro zahrnují možného zastínění 2 snímků, je každý bod z S_e^+ (S_e^-) rozšířen stejným způsobem do vyhledávací oblasti v dalším (předchozím) snímku a průměrná hodnota je minimalizována.

3.4.4 Hodnotící funkce „cost function“

IPAN Tracker používá hodnotící funkci uvedenou v [28]. Tato funkce je definována rovnicí (20).

$$\begin{aligned}\delta_s\left(P_{k-1,n}, P_{k,i}, P_{k+1,m}\right) &\\ = w_1\left(1 - \frac{\overline{P_{k-1,n}, P_{k,\iota}} \cdot \overline{P_{k,\iota}, P_{k+1,m}}}{\left\|\overline{P_{k-1,n}, P_{k,\iota}}\right\| \cdot \left\|\overline{P_{k,\iota}, P_{k+1,m}}\right\|}\right) &\\ + w_2\left(1 - \frac{2[\left\|\overline{P_{k-1,n}, P_{k,\iota}}\right\| \cdot \left\|\overline{P_{k,\iota}, P_{k+1,m}}\right\|]^{1/2}}{\left\|\overline{P_{k-1,n}, P_{k,\iota}}\right\| + \left\|\overline{P_{k,\iota}, P_{k+1,m}}\right\|}\right) &\end{aligned} \qquad (20)$$

Kde: $w_1 = 0.1$ první váha, $w_2 = 0.9$. První výraz je nastaven na 0, pokud je jeden z vektorů nulový.

$\overline{P_{k-1,n}, P_{k,\iota}}$ a $\overline{P_{k,\iota}, P_{k+1,m}}$ jsou vektory ukazující z $P_{k-1,n}$ na $P_{k,i}$ (z $P_{k,i}$ na $P_{k+1,m}$)

První výraz penalizuje změny ve směru, druhý ve velikosti vektoru rychlosti.

K této funkci jsou v [28] vázány následující předpoklady:

- Při inicializaci - žádné překrytí v F_1, F_2, F_3. Korespondence mezi F1 a F2 je zpočátku nastavena pomocí nejbližších sousedů. Později ji lze změnit.
- Strategie linkování – použit iterativní „greedy exchange“ algoritmus v dopředném i zpětném směru.
- Zvládnutí překrytí – překrytí je indikováno v F_k, když $N_{k-1}>N_k$. Poslední viditelný bod v rozbité trajektorii je nalezen v F_{k-1}, jako bod, který neuspěl při linkování pomocí extrapolace bodu z F_k.

V případě IPAN algoritmu, v sekci post-processingu je vyhledávací oblast S_e^+ získána používáním *limity hodnoty* δ_{max} na dva oddělené výrazy z hodnotové funkce (w1, w2).

První výraz tedy omezuje směr, jako $\theta \in [\theta_1, \theta_2]$. Druhý výraz, společně s limitem rychlosti v_{max}, omezuje rychlost jako $v \in [v_1, v_2] \cap [0, v_{max}]$. Jak již bylo zmíněno v 3.4.3, menší odchylka od hladkosti je pro hypoteticky zastíněný bod povolena, na rozdíl od aktuálně pozorovaného bodu. Matematicky je směrová kontinuita ztracené trajektorie vynucována omezováním výrazů rovnice (20). V rámci tohoto omezení je snadné odvodit, že:

$$\theta_{1,2} = \theta_e \mp \arccos(1 - \delta_{max}) \quad (21)$$

$$v_{1,2} = \frac{v_e(1 \pm \sqrt{\delta_{max}(2-\delta_{max})})^2}{(1-\delta_{max})^2} \quad (22)$$

Kde: θ_e je směr, v_e velikost přicházejícího vektoru rychlosti $\overrightarrow{v_e}$.

Z těchto rovnic lze určit, že maximální povolená změna směru ztracené trajektorie je $\pi/2$, když δ_{max} =1. Pokud je $\delta_{max} = 0.5$ limit změny je $\pi/3$.

- Limity rychlosti jsou $0 \leq v_1 \leq v_e$, zatímco $v_2 \geq v_e$.
- Pokud je δ_{max} nastavena blízko k hodnotě 1, $v_1 \approx 0, v_2 \gg v_e$
- Vyhledávací oblast S_e^- je získána podobně jako pro případ S_e^+.

3.4.5 Nastavení parametrů trasovacího algoritmu

Tento algoritmus má 3 důležité parametry, jejichž nastavení ovlivní trasování.

- Limit nejvyšší rychlosti v_{max} – jedná se o kritický parametr. Čím lepší bude odhad tohoto parametru, tím lepší bude výsledek (kvality, času). Pokud je rychlost *v* menší, než aktuální budou některé linky ztraceny, protože nebudou spadat do oblasti vyhledávání. Nastavením rychlosti na "bezpečnou" vysokou hodnotu také není správné, protože čas zpracování roste spolu s pravděpodobností přiřazení špatné linky.
- Hloubka ověřování n_{ver} - je vhodné nastavit na n_{ver} = 2 (přesnost).
- Limit hodnoty δ_{max} - záleží na charakteru pohybu. Pro plynulé pohyby může být nastaven relativně nízko. Pro povolení rychlých otočení a změny pohybu by měl být nastaven blíže k 1.

Při správném nastavení všech těžko parametrů by měl algoritmus být schopen sledovat body, které se mohou v obraze i dočasně ztratit a poté objevit.

3.5 Trasování pomocí sub-prostorového omezení, RC (rank constrained) [29]

Obvykle je trasovací problém řešen trasováním více jednotlivých významných bodů najednou, při použití trackeru jako například KLT, nebo některým z něj odvozeným. I když tento přístup funguje dobře při zpracování vysoce kvalitních videí a „silných“ bodů, občas selže, když se potýká s temnými a zašuměnými videosekvencemi obsahující nepříliš kvalitní body. Tato metoda je navržena pro společné trasování sady významných bodů, která umožnuje sdílení informací mezi různými významnými body ve scéně. Metoda může být použita pro trasování tuhých i pružných pohybů těles (případně páry pohybujících se těles) dokonce, i když je některý bod zastíněn. Kromě toho, může být použit k výraznému zlepšení výsledků trasování v nedostatečně osvětlených scénách (kde je mix silných a slabých bodů). Tento přístup nevyžaduje přímé modelování struktury, nebo pohybu scény.

3.5.1 Seznámení s problémem

Existuje několik reálných situací, kdy KLT a mnoho jeho alternativ nefunguje správně: slabé osvětlení, zašuměná videa, a když existují přechodné zastínění (occlusions), které by měly být ignorovány. S cílem vypořádat se s takovýmito případy více robustně, by bylo vhodné povolit významným bodům komunikovat mezi sebou (mezi jednotlivými body), k rozhodnutí jak se budou pohybovat jako skupina, tak aby respektovaly základní 3D geometrický pohyb ve scéně.

Tato základní geometrie omezuje trajektorie trasovaných bodů, aby měly strukturu s nízkou úrovní/hodností (low-rank). Tato metoda kombinuje low-rank geometrii skupiny trasovaných bodů spolu s úspěšným nelineárním trasováním jednotlivých bodů v rámci práce Lucas Canade, a to pomocí přidání low-rank regularizační penalizace pro optimalizaci trasování. Pro přizpůsobení dynamickým scénám s netriviálním pohybem je použito omezení s nízkou úrovní přes posuvné okno tak, abychom uvažovali pouze malý počet snímků za daný čas (to je společná myšlenka pro práci i s pružnými pohyby).

Při odhadu optického toku je cílem najít posunutí bodů, mezi po sobě jdoucími snímky, za předpokladu, že pole toku je lokálně téměř konstantní. I když je problém při trasování bodů podobný, nevyžaduje odhadování toku vynucováním omezení neměnnosti jasu. Nicméně, male chyby v optickém poli v každém snímku v tomto přístupu vedou k hromadění chyb v trajektoriích získaných integrací toku. Tyto chyby jsou při trasování nepřijatelné.

3.5.2 Trajektorie bodu s nízkou úrovní

V afinním modelu kamery by pro trajektorii jednoho bodu ze sady bodů z tuhých těles měl existovat afinní podprostor 3. dimenze, nebo lineární podprostor 4. dimenze. Nicméně podprostory odpovídající velmi degenerovaným pohybům mají nižší úroveň, než ty, které odpovídají obecným pohybům.

Uvažujeme dočasné posuvné okno, kde je při krátkém trvání pohyb jednoduchý a trajektorie bodu má nízkou úroveň. Omezení délky trajektorie bodu může také pomoci ve vyhovění pro aproximaci lokálního afinního modelu kamery ve scénách, které porušují afinní model kamery. Nakonec, dokonce i v případě více pohybující se tuhých objektů je sada trajektorií stále s nízkou úrovní.

3.5.3 Trasování bodu

Lokace významného bodu $z_1 \in R^2$ v daném $N_1 \times N_2$ snímku z $N_1 \times N_2 \times N_3$ videa je charakterizována šablonou $\boldsymbol{T}$, která je $n \times n$ pod-obraz tohoto snímku se středem v bodě z_1(n je malé celé číslo, obecně liché, takže šablona má centrální pixel). Pokud z_1 nemá celé (integer) souřadnice, $\boldsymbol{T}$ je interpolována z obrazu. Označíme $\Omega = \{1,..,n\} \times \{1,..,n\}$ a parametrizujeme $\boldsymbol{T}$ tak, že jeho pixelová hodnota je získána podle $\{T(u)\}_{u\in\Omega}$.

Klasická formulace problému trasování jednotlivých bodů spočívá v hledání translace x_1 takové, že minimalizuje vzdálenost mezi šablonou bodu $\boldsymbol{T}$ v daném snímku a dalším snímkem videa vypočteném dle x_1; označíme tento další snímek jako $\boldsymbol{I}$. To znamená, že minimalizujeme funkci energie jednotlivých významných bodů $c(x_1)$.

$$c(x_1) = \frac{1}{n^2} \sum_{u \in \Omega} \psi(T_{(u)} - I_{(u+x_1)}) \qquad (23)$$

Kde, například $\psi(x) = |x|$, nebo $\psi(x) = x^2$. Pro aplikování spojité optimalizace se koukáme na x_1 jako na spojitou proměnnou, a mohli tak pohlížet na $\boldsymbol{T}$ a $\boldsymbol{I}$ jako na funkce přes spojitou doménu (implementovanou s bilineární interpolací).

3.5.4 Regularizace s nízkou úrovní

Pokud chceme podpořit strukturu nízké úrovně v trajektorii, nemůžeme pohlížet na trasování rozdílných významných bodů jako na oddělený problém. Pro $f \in \{1,2 \dots, F\}$ označme x_f jako pozici významného bodu f v aktuálním snímku (v obrazových souřadnicích) a $x = (x_1, x_2, \dots, x_F) \in R^{2F}$ označíme jako společný stav všech bodů ve scéně. Můžeme definovat celkovou energii jako:

$$C(x) = \frac{1}{Fn^2} \sum_{f=1}^{F} \sum_{u \in \Omega} \psi(T_{f\,(u)} - I_{(u+x_f)}) \quad (24)$$

Kde: $T_{f\,(u)}$ je šablona pro bod *f*. Nyní můžeme stanovit požadované vztahy mezi významnými body ve scéně pomocí stanovení omezení na oblasti optimalizace rovnice (24). Namísto prosazování tvrdého omezení, my přidáváme penalizační termín k (24), který zvyšuje cenu stavů, které jsou v rozporu s pohybem s nízkou úrovní. Konkrétně definujeme:

$$\bar{C}(x) = \alpha \sum_{f=1}^{F} \sum_{u \in \Omega} \psi\left(T_{f\,(u)} - I_{(u+x_f)}\right) + P(x) \quad (25)$$

Kde: ***P*** (x) je odhad z aproximace, nebo aproximace pro dimenzi sady trajektorií bodů přes posledních několik snímků z videa (k minulým pozicím bodům se chováme jako ke konstantám, čili toto je funkce pouze současných stavů x). Faktor měřítka $\frac{1}{Fn^2}$ byl nahrazen konstantou α, protože tento koeficient je nyní také odpovědný za řízení relativní síly penalizačního výrazu.

Tato práce dává vzniknout 2 různým řešením, charakteristickým podle síly penalizačního výrazu α. Každá z nich má užitečné využití v real-time trasovacích aplikacích. V prvním případě, předpokládáme, že většina (ale ne nutně všechny) bodů ve scéně přibližně odpovídají modelu s nízkou úrovní. To je vhodné v případě, že scéna obsahuje netuhé, nebo více pohybujících se těles. Můžeme stanovit slabé omezení tím, že uděláme penalizační výraz nízký v porovnání s ostatními výrazy α_{weak}. Pokud je významný bod silný, bude s jistotou trasovat bod ignorováním omezení (bez ohledu na to, zda je pohyb v souladu s ostatními pohyby ve scéně). Pokud je bod slabý, ve smyslu, že nemůžeme zcela určit jeho skutečnou pozici, poté se stane penalizační termín více významný a bude podporovat bod, aby jeho pohyb souhlasil s pohybem ostatních bodů ve scéně.

Ve druhém případě (α_{strong}) předpokládáme, že všechny body ve scéně by měly souhlasit s modelem s nízkou úrovní (a odchylky od tohoto modelu jsou pouze orientační chyby trasování). Můžeme stanovit silné omezení uděláním penalizačního výrazu velkým v porovnání s ostatními výrazy. Žádná malá sada bodů nemůže přemoct omezení, bez ohledu na to, jak silné body jsou. To přinutí všechny body pohybovat se stejným směrem, který je v souladu s jednoduchými pohyby. Tak může být malý počet bodů dokonce na chvíli ztracen, a jejich pozice bude predikována pomocí pohybu ostatních bodů ve scéně.

3.5.5 Konkrétní volby nízko-úrovňové regularizace

Omezíme se na ukázání výsledku s použitím 2 možností pro $\boldsymbol{P}$ popsaných níže. Každá volba definuje $\boldsymbol{P}$ (x) ve výrazu jako matici $\boldsymbol{M}$. Je to $2(L+1) \times F$ matice, jejíž f -tý sloupec obsahuje trajektorii bodu pro bod f uvnitř posuvného okna pro L+1 po sobě jdoucích snímků (současný a L minulých snímků).

Konkrétně, $M = [m_{i,j}]$, kde $(m_{0,f}, m_{1,f})^T$ je současná (proměnná) pozice bodu f a $(m_{2l+1,f}, m_{2l+2,f})^T$, kde $l = 1,2 \ldots L$ obsahují x a y souřadnice pixelu bodu f z předchozích l snímků (k minulým pozicím bodu se chováme jako ke konstantě). Jednou možnou alternativou je soustředit/centrovat sloupce z $\boldsymbol{M}$ pomocí odčítání průměrné hodnoty všech sloupců z každého sloupce. Většina omezení odvozených pro trajektorie (za předpokladu například tuhých pohybů) ve skutečnosti omezují trajektorie na nízko-úrovňový afinní podprostor (narozdíl od lineárního podprostoru). Centrování sloupců z $\boldsymbol{M}$ transformuje afinní omezení na lineární. Druhou alternativou je možnost upustit od centrování a pohlížet na afinní omezení jako na lineární omezení o jednu dimenzi vyšší.

3.5.5.1 Explicit Factorizations

Jednoduchou metodou pro vynucení omezení struktury je napsat $\boldsymbol{M} = \boldsymbol{BC}$, kde $\boldsymbol{B}$ je $2(L+1) \times d$ matice, a $\boldsymbol{C}$ je $d \times F$ matice. Nicméně, jak bylo zmíněno v předešlé sekci, protože trasovaný bod často neleží přesně v podprostoru kvůli odchylkám z modelu kamery, nebo netuhosti, konkrétní omezení v této podobě není vhodné. Nicméně explicitní faktorizace může být použita v penalizačním výrazu pomocí měření odchylky od $\boldsymbol{M}$ v nějaké normě, z její aproximace faktorizace nízké úrovně (low rank faktorizace). Například, pokud

$$M = U\Sigma V^T \qquad (26)$$

Označíme (4) jako SVD z $\boldsymbol{M}$ (singular value decompozition), můžeme stanovit:

$$P(x) = ||\boldsymbol{BC} - \boldsymbol{M}||_* \qquad (27)$$

kde: $\boldsymbol{B}$ jsou první 3, nebo 4 slupce z $\boldsymbol{U}$, $\boldsymbol{C}$ jsou první 3, nebo 4 řádky z ΣV^T. Poté toto $\boldsymbol{P}$ koresponduje s penalizační maticí $\boldsymbol{M}$ skrz $\sum_{i=d+1}^{F} \sigma_i$, kde $\sigma_i = \Sigma_{ii}$ je i-tá singulární hodnota z $\boldsymbol{M}$. Protože historie je pevná U, Σ, V^T jsou funkce pouze x.

Tento přístup předpokládá znalost o hodnosti (low-rank) $\boldsymbol{d}$. Pro jednoduchost předpokládáme blízký tuhý model a tak nastavujeme $\boldsymbol{d} = 3$ pokud centrujeme M a $\boldsymbol{d} = 4$ pokud necentrujeme.

3.5.5.2 Nuclear norm

Populární alternativou k explicitnímu udržení trasování nejlépe vyhovujícího nízko-úrovňového podprostoru k M je použít nukleární normu matice:

$$P(x) = ||M||_* = ||\sigma||_1 \quad (28)$$

Tohle je konvexní aproximace pro hodnost z M.

Zde $\sigma = (\sigma_1\ \sigma_2\ ...\ \sigma_{2(L+1)^F})^T$ je vektor singulárních hodnot z M, a $||.||_1$ je l1 norma. Na rozdíl od explicitní faktorizace, kde je penalizována pouze energie mimo prvních d principiálních komponent z M, nukleární norma upřednostňuje M s nízkou hodností před M s vysokou hodností dokonce i když obě matice mají hodnost ≤d. Tak, použitím tohoto druhu penalizace budou upřednostňovány jednodušší pohyby trasovaných bodů před komplexnějšími, dokonce, i když jsou oba technicky přípustné.

3.5.6 Detaily implementace

Funkce $\psi(x)$ v rovnici (25) byla ponechána ve tvaru $\psi(x) = |x|$. Tato forma pro ψ byla použita pro celou rovnici, takže všechny výrazy v celkové funkci energie se chovají lineárně ve známém rozsahu hodnot. Byl stanoven pevný parametr m pro každou formu penalizace (zvolené empiricky – viz doplňkový materiál autorů článku v příloze), který určuje sílu penalizace. Silné a slabé regularizační parametry jsou nastavený jako: $\alpha_{weak} = \frac{1}{mn^2}$ a $\alpha_{strong} = \frac{1}{mFn^2}$

Slabý parametr měřítka (α_{weak}) znamená, že perfektně shodující se body přispějí nulou k totální energii, a slabě shodující se body přispějí částkou v řádu 1/m k celkové energii. Penalizační výraz bude přispívat v řádu jednotek k celkové energii. Protože nemůžeme dělit příspěvky každého bodu počtem všech bodů, příspěvky penalizačního výrazu jsou srovnatelné velikostí jednotlivých bodů.

Silný parametr měřítka (α_{strong}) naznačuje, že penalizační výraz je ve stejném měřítku, jako součet příspěvků ze všech bodů ve scéně.

Ke zhodnocení metody autoři provedli testy na několika reálných videosekvencích za podmínek, které jsou obtížné pro trasování. Ty zahrnují roztřesené záběry ve špatných světelných podmínkách. Výsledná videa obsahují tmavé místa s pár dobrými body a nestabilní pohyb kamery se špatným osvětlením zavadí časově proměnné pohybové zkreslení.

4 NÁVRH KRITÉRIA HODNOCENÍ, EXPERIMENT PRO REALIZACI

Cílem této kapitoly je vytvoření experimentu, který bude zahrnovat testování jednotlivých trasovacích algoritmů, zejména jejich vlastností a schopností si poradit s různými situacemi (zmizení bodu/objektu, překrytí bodu/objektu, změna osvětlení a další). Další částí této kapitoly bude navrhnutí kritéria, podle kterého bude hodnocena úspěšnost těchto algoritmů. Nejdůležitějším kritériem bude měření odchylky od trasovaného významného bodu, poté může být hodnocena schopnost nalezení nových vhodných bodů, jejich udržení atd. Po definici tohoto kritéria a pořízení snímků k jeho realizaci budou zvolené algoritmy implementovány a srovnány z hlediska tohoto kritéria.

4.1 Návrh kritéria hodnocení

Navržené kritérium bude ovlivňovat výsledné hodnocení jednotlivých trasovacích algoritmů. A na základě tohoto kritéria bude také celkově zhodnocena kvalita těchto algoritmů. Jak již bylo uvedeno v předešlé kapitole, účelem trasování je zaznamenávání trajektorií pohybu významných bodů neboli tras. V podstatě se jedná o zaznamenání veškerých pozic, kterými bod prošel. Poloha sledovaného bodu může být vyjádřena například pozicí prostředního pixelu vyhledávacího okénka v obraze. Tyto souřadnice se mohou ukládat do řetězce, který obsahuje veškeré pozice, kterým pozorovaný bod prošel, přičemž každý bod má svůj řetězec (svou trasu). Tyto řetězce lze vzájemně pozorovat.

Pokud bychom tedy měli určitě referenční a správně označené trajektorie (respektive posloupnost středových bodů), bylo by možné porovnávat jednotlivé trajektorie různých trasovacích algoritmů a určit jejich odchylky a vzdálenosti těchto odchylek vzhledem k referenční trajektorii. Pokud bychom tedy od začátku měli definované body, které chceme sledovat, můžeme porovnávat mezi sebou schopnosti trasovacích algoritmů pro sledování bodů. Tyto body lze definovat v několika úvodních snímcích videosekvence, například při použití Shi & Thomasi detektoru, případně jiných detektorů významných bodů. Po získání těchto bodů je již možné je ve videosekvenci sledovat. Jedno z kritérií pro hodnocení algoritmů tedy bude odchylka od referenční trajektorie. Referenční trajektorie bude určena tak, že budou z prvních snímků nalezeny body vhodné pro trasování, a poté postupně pro každý snímek pevně a ručně označeny a definovány. Odchylka od této referenční trajektorie může být definována pomocí různých kritérií, než jen pouhou vzdáleností mezi jednotlivými středy.

Tato kritéria lze řadit pomocí zvolených metrik používaných v ortogonálním rastru, ty jsou definované pomocí Minskowského vzdálenosti. Ta obsahuje typy vzdáleností jako např. Chebyshev, Cityblock a Euklidovskou vzdálenost, která bude právě použita pro ohodnocení vzdálenosti mezi trasovaným a referenčním bodem. Jedná se o součet kvadrátu odchylek vzdálenosti definovaných jako:

$$d^{L_2}(x,y) = \sqrt{\sum_i (r_i - s_i)^2} = \sqrt{(x_{ref} - x)^2 + (y_{ref} - y)^2} \qquad (29)$$

Dle této vzdálenosti bude stanovena velikost odchylky trasovaného bodu.

Velmi významným parametrem pro algoritmy je robustnost trasování, která bude hodnocena a souvisí s odolností algoritmů vůči změnám podmínek. Mezi tyto změny lze zařadit geometrické transformace, jako jsou například translace, rotace kamery a změna měřítka. Tato odolnost totiž také ovlivňuje schopnost algoritmů sledovat významné body. Především při změně měřítka může docházet ke ztrátě trasovaného bodu případně k velké odchylce od pozice. Také změna prostředí při snímání má velký vliv na kvalitu trasovacího algoritmu a odolnost vůči těmto změnám je žádoucí. Mezi takové změny lze počítat především změnu osvětlení scény, ať již osvětlení místnosti pomocí světel, nebo i osvětlení venkovního prostředí (slunce, lampy). S tímto osvětlením je také spojený vznik stínů, které mohou vrhat objekty (pohyblivé, statické stíny). Další problém, jehož zvládnutí zvyšuje robustnost algoritmů, je schopnost odolávat šumu, který může být způsobený buď kamerou (vadné buňky snímače), nebo může být způsobený pohybem kamery (již zmiňovaný motion blur), což může být problém v závislosti na rychlosti pohybu pro případ ručně držené kamery. Dalším parametrem, který může působit potíže, je změna vzhledu bodů/objektu v průběhu trasování. Tato změna může být způsobená například pouhou rotací objektu (např. rotace míče), deformací objektu (při nárazu) a další. Úspěšné zvládnutí všech možný působících změn je obtížné, ovšem pokud si algoritmus dokáže s těmito situacemi poradit, bude velmi robustní a jeho použitelnost bude vysoká.

Dalším kritériem, kterým lze hodnotit trasovací algoritmy, může být schopnost správně nalézt a udržet si trasovaní bodů v průběhu videosekvence. Tato vlastnost souvisí se stabilitou trasování, jelikož správné nalezení a detekce konkrétního bodu přes celou videosekvenci značí, že algoritmus je spolehlivý. Pokud ovšem dojde ke ztrátě trasovaného bodu (chybou algoritmu, překrytím bodu/objektu, změnou podmínek), je důležité také jak dlouho je algoritmus schopen významný bod sledovat. Tato doba také souvisí se stabilitou, zejména pro dlouhé videosekvence, kde je vhodné, aby byl trasovaný bod udržený co nejdéle.

Jinou možností hodnocení kvality algoritmů je sledovat počet ztracených bodů, který souvisí s robustností a stabilitou. Pokud totiž nebude trasovací algoritmus odolný vůči změnám, nebo bude nepřesný, dojde ke ztrátě bodů při trasování. Mezi možnosti, které mohou vést ke ztrátě bodů, patří trasování nevhodných bodů (falešná detekce, zastíněný bod). Dále trasování bodů, které se ukázaly být nestabilní pro videosekvence (krátká doba trasování, velké vzdálenosti od referenčních trasovaných bodů). Také body, které byly ztraceny například kvůli překrytí překážkou, nebo kvůli velké změně tvaru objektu a již se nepodařily znovu nalézt. Případně kvůli šumu, špatnému osvětlení nebo špatné kvalitě videosekvence. Další možností, díky které lze ztratit body, je při prudké změně směru pohybu bodu/objektu nebo kamery, jelikož v tomto případě může dojít k špatné predikci následujícího stavu a tím pádem vyhledávání bodu v místech, kde se tento bod nenachází, což vede tak ke ztrátě toho bodu. Pro toto kritérium nebude důležité, při které události došlo ke ztrátě bodu, ale kolik bodů celkově bylo v průběhu trasování ztraceno. Tento počet bude vyjádřen procentuálně.

Kritérium, které lze hodnotit u trasování, je také schopnost algoritmů znovu nalézt bod, který byl ztracený. Většinou je možné ztracený bod najít, pokud se v obraze po pár snímcích bod znovu objeví (např. kvůli překrytí sloupem) na předpokládaném místě. Takový bod mohou najít algoritmy, které pracují s trajektoriemi bodu/objektu, bod ovšem nesmí být ztracený příliš dlouho. Tato schopnost je výhodná, pro pozorování daného objekt celou dobu trasování.

Posledním důležitým kritériem, které bude hodnoceno u vybraných algoritmů, je jejich výpočetní rychlost (náročnost), jedná se o parametr, který je důležitý především pro nasazení aplikací v praxi, případně pro real-time zpracování. Jedná se o dobu, která je nutná ke zpracování jednoho snímku, neboli jde o parametr, který určuje, jakou maximální frekvenci snímání algoritmus umožňuje.

Výsledná kritéria tedy jsou:

- Odchylka trasovaného bodu od referenčního (pomocí kvadrátu odchylek)
- Odolnost algoritmů vůči změnám podmínek = robustnost (osvětlení, stín, motion blur, změna vzhledu objektu, geometrické transformace, drift)
- Počet a doba úspěšně trasovaných bodů = stabilita
- Procentuální počet ztracených bodů (změny podmínek, prudká změna směru pohybu kamery, objektu)
- Schopnost znovu nalézt a trasovat ztracený bod (např. při překrytí)
- Výpočetní náročnost algoritmů

4.2 Návrh experimentu pro realizaci trasování

Cílem toho experimentu bude ověřit a porovnat vlastnosti vybraných trasovacích algoritmů a na základě výsledků těchto experimentů algoritmy zhodnotit. Experiment je rozdělen do dvou fází.

V první fázi se bude testovat hlavně schopnost algoritmů udržet a trasovat daný bod a bude zkoumána odchylka od vyznačených referenčních bodů a to v přítomnosti geometrických transformací (translace, rotace, změna měřítka). Dále bude zkoumána i stabilita algoritmů (počet a doba úspěšného trasování).

V druhé fázi poté budou testovány dohromady ostatní vlastnosti algoritmů, především robustnost (odolnost algoritmů vůči změnám) a schopnost znovu nalézt ztracený bod.

V obou případech testování bude vyhodnocen procentuální počet ztracených bodů a také výpočetní rychlost algoritmů. To vše při natáčení videosekvence z rukou držené kamery. Pro tento experiment bude použito více kamer s rozdílnými parametry (nastavení doby expozice, počet snímků/s, rozlišení atd.). Tyto parametry jsou přehledně zobrazeny v **Tabulka 3**.

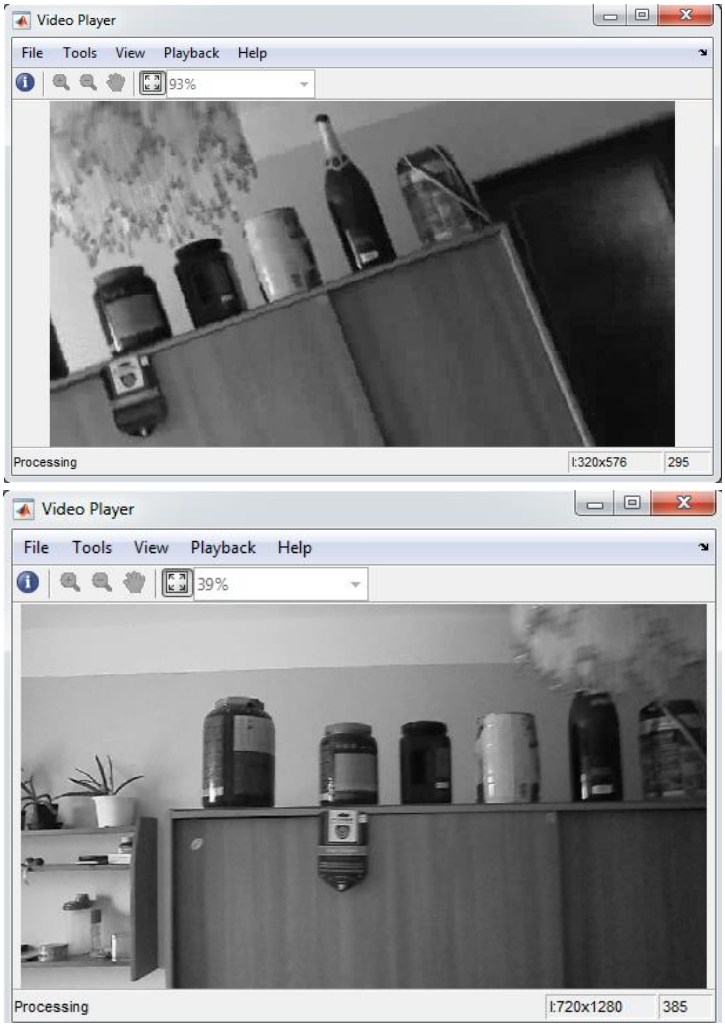
Zařízení	Doba expozice	Snímků/s	Rozlišení
Fotoaparát mobilního telefonu HUAWEI Y530 (MOBIL)	Automatická	30	320x576
Fotoaparát Olympus VG-130	Automatická	30	720x1280

Tabulka 3: Parametry použitých zařízení k tvorbě testovacích videosekvencí

Doba expozice byla nastavena na automatickou (proměnnou) hodnotu. Pro trasovací algoritmy je jednodušší se vypořádat s šumem, který je způsoben nízkou hodnotou doby expozice (většinou bodové znehodnocení pixelu), než s pohybovým zkreslením způsobeným dlouhou dobou expozice (celkové rozmazání a znehodnocení obrazu). Automatická doba expozice je volena jako kompromis mezi šumem a zkreslením pohybem. Nyní se již dostáváme k popisu obou fázi experimentu.

4.2.1 Test odchylek od referenčních bodů + (přesnost, stabilita)

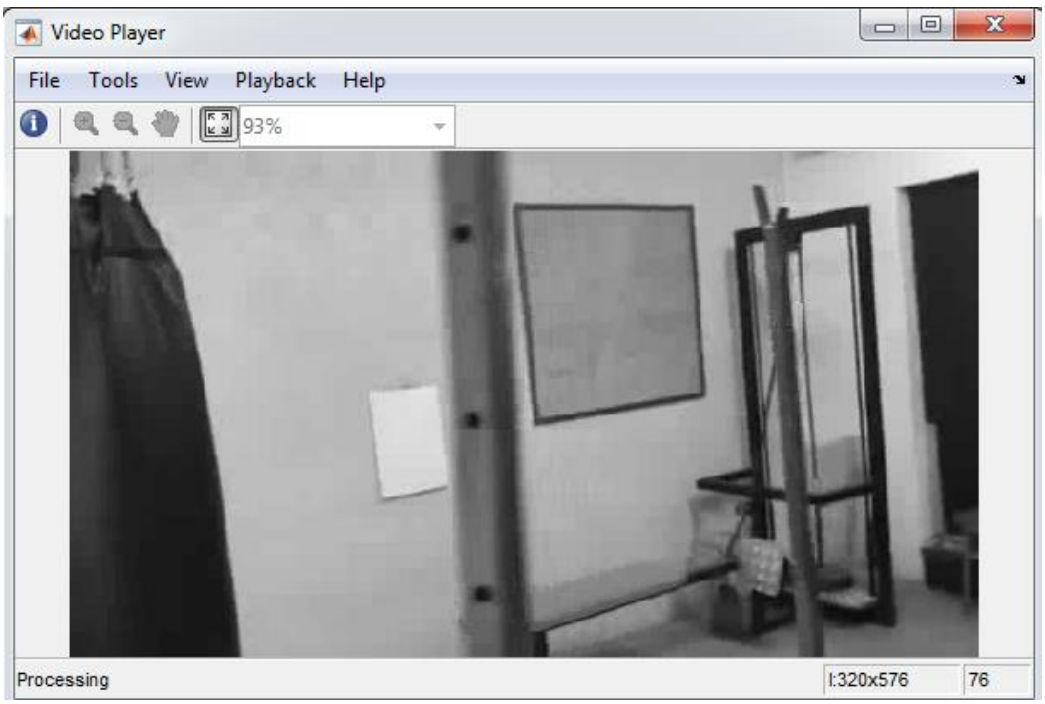
Jak již bylo zmíněno výše, videosekvence pořízené pro tento test slouží především k ověření schopnosti udržet nalezený bod po celou dobu trasování (pokud je k dispozici), dále k určení přesnosti s jakou je bod trasován (odchylka od referenčního bodu) a také je zde testován na odolnost vůči geometrickým translacím. V první části videosekvence je trasovací algoritmus testován pro translaci (pohyb zleva doprava), poté je testován vůči změně měřítka (odstoupení od objektu), následně rotace kolem osy kolmé na osu objektivu (otečení s kamerou doprava a zpět) a poté rotace kamery (natočení kamery o 90° a zpět). Nakonec je testována opět translace (zprava doleva), ale zároveň schopnost algoritmů nalézt body, které byly předtím ztraceny (avšak nikoliv kvůli překrytí, ale kvůli ztrátě bodu z dosahu snímání). Ukázka obrázků z této videosekvence je zobrazena na Obrázek č. 9. Horní obrázek je snímek z mobilního telefonu, spodní obrázek zachycuje snímek z fotoaparátu.



Obrázek č. 9 Snímky z průběhu videosekvence

4.2.2 Test robustnosti (odolnosti vůči změnám, chybám atd.)

Tato druhá videosekvence slouží k testování robustnosti algoritmů a jejich odolnosti vůči různým změnám. Na začátku videosekvence se testuje schopnost algoritmů znovu nalézt ztracené body a jejich správná detekce. K tomuto ověření slouží překrytí pomocí dvou kovových tyčí, které mají stejnou velikost, ale jsou od sebe vzdáleny tak, aby se zdály jinak široké (jedna tyč blíže ke kameře, druhá tyč vzdálena od kamery). Různá vzdálenost tyčí slouží především k testování toho, zda si algoritmy dokáží poradit se zakrytím, které překrývá významný bod ve 3 a více snímcích (bližší tyč), případně zda zvládnou opět nalézt bod, který je překrytý přes jeden nebo 2 snímky (vzdálenější tyč). Následně je po krátkém přesunu testována odolnost vůči šumu, přesněji vůči pohybovému zkreslení, které je simulováno rychlými pohyby kamerou do stran. Při těchto rychlých pohybech dochází k celkovému rozmazání obrazu, které je testováno pro nalezení bodů (i přes rozmazání). V další části je testována odolnost algoritmů vůči náhlé změně světelných podmínek, která obnáší vypnutí a následné zapnutí světel, kdy by měl být algoritmus schopný správně detekovat trasované body i navzdory těmto světelným vlivům. Nakonec je testována schopnost algoritmů vypořádat se s postupnou změnou světelných podmínek, která je simulována postupným pohledem do okna, na které svítí slunce přes závěsy. Jak lze vidět, tak je na každém snímku jiná úroveň osvětlení při postupném pohledu do okna a je testováno, zda algoritmy dokáží i přes tyto postupné změny udržet trasované body. Ukázky z této videosekvence lze vidět na Obrázek č. 10



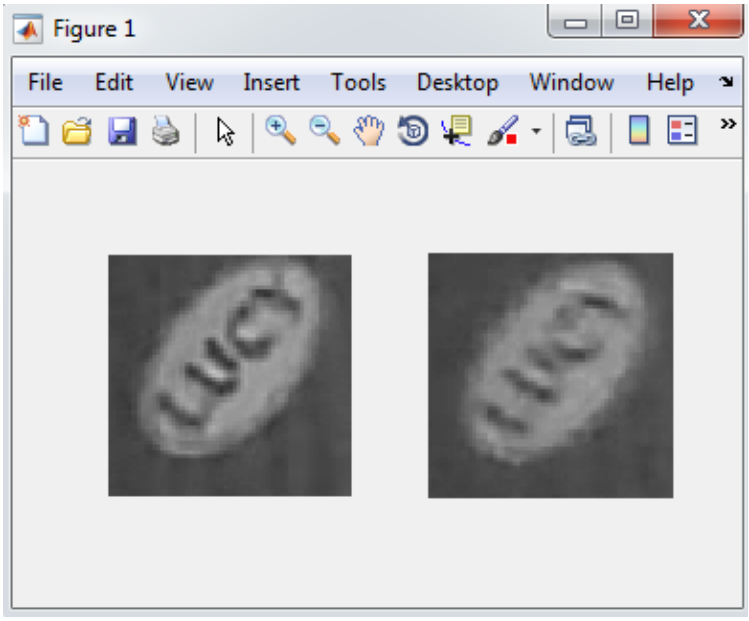
Obrázek č. 10 Ukázka z videosekvence pro testování robustnosti

4.3 Ukázka vlivu pohybového zkreslení na videosekvenci

Tato část je věnována důkazu o přítomnosti pohybového zkreslení v testovacích videosekvencích. Pro potvrzení o výskytu MB byla zvolena první testovací videosekvence, která byla pořízena fotoaparátem a na které lze vliv MB nejlépe i okem pozorovat. Pro tyto účely byly vybrány dva po sobě jdoucí snímky z průběhu natáčení a na oba tyto snímky (ostrý i zašuměný) byl aplikován detektor významných bodů s cílem nalézt stejné body na obou snímcích. Tyto snímky se nachází na Obrázek č. 11 (horní i spodní obrázek).

Obrázek č. 11 Ukázka vlivu MB na 2 po sobě jdoucí snímky

Na první obrázku je vidět 10 nalezených významných bodů, ovšem hned na následujícím snímku, na který byl aplikován detektor významných bodů (se stejnými parametry), lze pozorovat, že spousta bodů již nemohla být detekována na stejném místě. Tudíž byly tyto body nalezeny na jiných vhodných oblastech (body jsou označené červeně), body které byly správně nalezeny na obou snímcích, jsou označeny zeleně. To, že se na následujícím snímku body ztratily, je výsledek pohybového zkreslení, které lze vidět jak pouhým okem při pozorování změn obrazu a také při přiblížení v okolí významného bodu. Pro detailní pohled, jak působí pohybové zkreslení na snímek, jsem vybral bod, který je v obrázku označen číslem **1.** Při detailním přiblížení na tento významný bod, které je ukázáno na Obrázek č. 12 lze pozorovat, že na levém (ostřejším obraze, kdy ještě došlo k pohybu a tím ani k vlivu MB) lze vidět nápis „LUCY“, ovšem na druhém obrázku, který je zobrazen vpravo (snímek následující, na kterém došlo k pohybu kamery a tím i k MB) lze vidět, že nápis je již tak rozmazaný, že jeho přečtení je téměř nemožné. Tato vlastnost tedy způsobí rozprostření pixelů po okolí kolem textu, a tím i kolem významného bodu, což nevede k tomu, že na tomto snímku již nelze detekovat stejný bod.



Obrázek č. 12 Detailní vliv působení MB na okolí významného bodu

MB je způsobené pohybem, nebo spíše posunem kamery (v tomto případě doprava) v průběhu doby expozice, což má znatelný a viditelný vliv na celkové rozmazání obrazu, hlavně potom okolí významného bodu, což může způsobit ztrátu tohoto bodu. S tímto problémem se také potýká trasování významných bodů a předpokládá se, že toto pohybové zkreslení nezpůsobí ztrátu významného bodu ve snímku v žádné trasovací metodě, jelikož by měl být trasovací algoritmus vůči tomuto šumu odolný.

4.4 Implementace zvolených algoritmů a jejich popis

Tato sekce je věnována popisu implementovaných metod. Každá metoda bude podrobně popsána svojí funkčností i spolu s přidanými funkcemi, který byly vytvořeny v průběhu implementace jednotlivých metod. Všechny 3 implementované algoritmy byly vytvořeny v prostředí Matlab. Všechny metody mají společnou první část, která se skládá z *inicializace* (a vytvoření detektoru významných bodů dle zadaných parametrů), kdy se pro začátek trasování naleznou v prvním (pro IPAN v druhém) snímku významné body (metodou minima vlastní hodnot), které slouží jako počáteční body jednotlivých trajektorií. V prvním kroku tedy nastavíme parametry detektoru (maximální počet bodů, potlačované okolí, prahy) a nalezneme významné body v prvním (druhém) snímku. Další částí je *inicializace videopřehrávače*, ve kterém budou zobrazeny veškeré nalezené body. Přehrávání videosekvence probíhá ve while smyčce do doby, než je videosekvence na posledním snímku. Po přehrání videosekvence ještě probíhá *postprocessing*, který poté ještě pracuje s nalezenými body. V případě IPAN metody je v postprocessingu ještě implementována funkce na zpětné dohledání ztracených bodů, jak je popsáno v sekci Post-processing ztracených trajektorií a bude ještě probráno dále. Kromě tohoto zpětného dohledání ztracených bodů mají všechny algoritmy společné funkce, které slouží k porovnání s referenčními body. K tomuto porovnání slouží vytvořené funkce *DeviatonOfRefer*, která na základě zadaného poloměr potlačovaných bodů (vzdálenost, ve které by se neměl nacházet žádný jiný bod, pokud je v té oblasti již nějaký bod nalezen) nalezne nejbližší bod, jehož vzdálenost nepřesahuje daný poloměr (jinými slovy bod, který je nejblíže referenčnímu bodu). Ostatní body, které nejsou v blízkosti poloměru a nelze je tak s referenčními porovnávat mají hodnotu *NaN*. Po získání všech bodů, které lze srovnávat je spočítána odchylka každého bodu od svého referenčního bodu. Tato odchylka je jedním z hlavních hodnotících kritérií a bude v další sekci ukázána a porovnávána. Funkce poté vrací trajektorii, která odpovídá bodům, které jsou blízké referenčním a ostatní body jsou nulovány. Na základě těchto odchylek poté bude vyhodnocena schopnost metody udržet si své stabilní body (pokud budou body tzv. „driftovat“ bude jejich stabilita výrazně horší, než u ostatních metod, protože trasovací metody by měly svůj bod držet v nejlepší pozici). Toto je seznam funkcí, které jsou společné pro všechny 3 algoritmy. Kromě těchto funkcí má poté každý algoritmus své vlastní funkce, které budou popsány. Veškeré nalezené a trasované body se poté ukládají do proměnné *point_overview*.

4.4.1 Implementace metody KLT

Metoda KLT patří i v současné době mezi jednu z nejpoužívanějších a přitom stále spolehlivou a vcelku robustní metodu. Díky velké rozšířenosti tohoto algoritmu se pro nás stává spíše referenčním, se kterým budeme porovnávat zbylé 2 metody, i když budou srovnávány všechny metody společně. Jelikož je tato metoda implementována i jako funkce v matlabu je zde také používána. Pro použitím této metody si nejdříve musíme definovat třídu tohoto trackeru a definovat jeho parametry, mezi které patří: *maximální počet pyramid, maximální chyba v dopředném i zpětném směru, velikost bloků (rozměry šablony), maximální počet iterací* veškeré tyto parametry byly pro trasování ponechány na svých původních hodnotách. Prvním krokem pro trasování tímto algoritmem byla inicializace trackeru pomocí funkce *initialize*, do které vstupovaly body nalezené v prvním snímku a také první snímek. Funkce slouží k započetí trajektorie bodů, od kterých se začne trasovat. Po inicializaci tohoto trackeru nastává samotné trasování významných bodů, ke kterému slouží příkaz *step,* ve kterém jsou jako parametry následující snímek a body nalezené v předchozím snímku, které jsou zaznamenány přímo v objektu *pointTracker*. Výstupem této funkce jsou nalezené body v novém snímku, a také proměnná *isFound*, která říká, které body se nepodařilo trasovat (ztracený bod). Pokud tedy dojde ke ztrátě trasovaného bodu, jsou pomocí detektoru dle minima vlastních hodnot nalezeny další významné body. Jelikož funkce obsahuje maximální počet bodů, které lze trasovat a také potlačovaný poloměr, je použita vytvořená funkce *findbestpoint3*, která má jako své vstupní proměnné tyto 2 parametry a také počet ztracených bodů při trasování a nově nalezené body pomocí detektoru spolu s jejich „sílou“ metrikou. Úkolem této funkce je nalézt body, které nezasahují do okolí kolem významných bodů definovaného poloměrem *radius* a zároveň jsou nejsilnější (mají nejvyšší hodnost metriky), aby bylo zajištěno co nejkvalitnější trasování. Funkce nejprve ve for cyklu hledá body (adepty), které splňují předpoklad správné vzdálenosti od již nalezených bodů. Po nalezení těchto vhodných adeptů jsou nově nalezené body seřazeny podle jejich síly. Pro nejsilnější body poté opět probíhá kontrola vzdálenosti, ale tentokrát mezi jednotlivými nalezenými body (ne starými trasovanými), kvůli tomu, abychom neměli několik nejsilnější bodů umístěných hned vedle sebe, ale v definované (prázdné) oblasti. Poté jsou tyto body vybrány jako nejlepší nalezené body. Takto nalezené body poté nahradí místo, kde došlo ke ztrátě bodu a použije se funkce *setPoint*, do které vstupují body, které byly nalezeny při trasování a také body, které nahradily místo po ztracených bodech, poslední parametr je *pointValidy*, který oznamuje, které body mají být trasovány. Celý tento algoritmus pracuje v hlavní smyčce videopřehrávače.

4.4.2 Implementace metody IPAN

Tato metoda je založena na trajektorii významných bodů, kterou ovlivňuje „cost“ funkce, která penalizuje změny ve směru a rychlosti bodu. Na tuto metodu má velký vliv pohybové zkreslení, protože je nutné v každém snímku nalézt nové body, které slouží jako vstupy do této funkce. Tento problém je však v metodě řešen postprocessingem, o jehož průběhu se ještě zmíníme. Jako u každé z vybraných metod se i u této začíná procesem *inicializace*, kde se detekují počáteční body. U této metody se detekují významné body až na 2. snímku jak bylo uvedeno v teoretické části.

Poté je použita funkce *FindAllCandidateInital*, která má jako vstupní parametry maximální počet trasovaných bodů, maximální počet kandidátů, snímek ve kterém se kandidáti hledají, nalezené body, práh pro detektor a odhad maximální rychlosti, a jako poslední příznak *initial*, který signalizuje, že se hledají body v 1. a 3. snímku pro zajištění počátečních kandidátů v dané oblasti *Vmax* kolem bodů nalezených ve druhém snímku. Uvnitř této funkce se nejprve pro každý jednotlivý bod nalezený buď ve druhém snímku (*initial*=1), nebo prvním, druhém (*initial*=0) snímku nalezena oblast ROI, která slouží k ohraničení oblasti, ve které se hledají významné body na daných snímcích (jako parametry funkce *FindAllCandidateInital*).

Hledání této oblasti probíhá ve vytvořené funkci *FindROI*, která ze zadaného poloměru (Vmax) a nalezených bodů v předchozím snímku vypočítá oblast v obraze, ve které se budou hledat významné body, tato funkce obsahuje i ošetření okrajů obrazu (malá/velká šířka, malá/velká výška a všechny jejich kombinace). Výstupem funkce *FindROI* jsou tedy souřadnice v obraze, které představují oblast hledání. Poté jsou v této vymezené oblasti nalezeny významné body dle detektoru minima vlastních hodnot s prahem. Tyto body jsou také zároveň výstupem z funkce *FindAllCandidateInital*. Pokud již máme nalezeny tyto počáteční body, je tato funkce aplikována ještě celkem 4x pro směr od prvního obrazu ke třetímu (aplikace na 2. a třetí obraz) a obráceně vše s hodnotu *initial*=0. Jediný rozdíl je v tom, že oblast je počítána pro každý nalezený bod z daného okolí a pro každý bod se vypočítá (a tím i rozšiřuje) oblast hledání. Jako příklad můžeme předpokládat 5 nalezených bodů v prvním snímku, při hledání kandidátů ve druhém snímku může být pro každý bod nalezeno například také 5 bodů v dané oblasti (pro druhý snímek tedy v tomto případě 25 bodů) a pro třetí snímek může být pro každý bod také nalezeno 5 bodů (možnost nalezení až 125 bodů). Většinou však 2 kandidáti pro každý bod. Počet kandidátů je určován jako parametr této funkce.

Poté je použita funkce *PointSumInitial*, která má za úkol sloučit veškeré nalezené body v daných snímcích pro danou oblast a zároveň eliminovat body, které jsou detektorem nalezeny víckrát (pomocí funkce unique) všechny vytříděné a sjednocené body pro daný snímek v daných oblastech jsou nastaveny jako výstupní parametry této funkce. Dalším výstupním parametrem je počet nalezených bodů v jednotlivých snímcích (F1-F3) pro dané oblasti (1:*MaxNrTrackedPts*), jako poslední výstupní parametr je počet kombinací pro dané oblasti, kterými lze kombinovat veškeré nalezené body, aby z nich byla jedna trajektorie, například pokud budou v dané oblasti nalezeni 2 kandidáti, máme 2^3 možných kombinací bodů, tak aby pokaždé tvořily jedinou trajektorii.

Dále v programu je for cyklus, který pokud v dané oblasti není nalezen další bod, tak v tomto místě uměle navýší hodnotu na 1, aby mohly tyto parametry pokračovat do následující funkce, ve které se již počítá „cost“ funkce, která tyto body nakonec sice stejně vyřadí, jelikož nebude nalezena vhodná trajektorie, ale také se v tomto for cyklu již začínají počítat ztracené body.

Nyní slouží vypočtené parametry jako vstupy do funkce *findbestcandidateinitial*, kdy jsou v celkem 4 for cyklech (3 pro každý snímek F1-F3, 1 pro danou oblast) skládány veškeré možné trajektorie, které slouží jako vstupy pro funkci *TestHypotesis* spolu s nastavením vah w1, w2 uvedené v teoretické části. Tato funkce implementuje „cost“ funkci penalizující změny trajektorie. Výstupem této funkce je pak číselná hodnota „ceny“ jednotlivých trajektorií. Po skončení 3 for cyklů (pro všechny snímky F1-F3) získáme hodnoty „ceny“ všech kombinací trajektorií v dané oblasti. Dále uvnitř funkce *findbestcandidateinitial* tedy hledáme takovou trajektorii, která má nejnižší „cenu“ a tu poté označíme jako začátek trajektorie. Funkce dále ošetřuje případy, kdy není nalezena žádná vhodné trajektorie. Výstupem funkce je poté seznam indexů, které říkají, jaká kombinace je nejvhodnější pro danou oblast.

Po získání této kombinace se ve for cyklu vytvoří a uloží nalezená optimální trajektorie, a zaznamenají se body, které byly ztraceny, jelikož se nepodařilo sestavit odpovídající trajektorii. Nalezené body se poté ukládají do proměnné s přehledem všech bodů, ztracené do proměnné *B_body*.

Nyní se již dostáváme do smyčky videopřehrávače, ve kterém se na začátku každé iterace získávají body z minulého snímku a body ze snímku před tímto snímkem (minulý a předminulý). Pokud došlo v minulém snímku ke ztrátě bodu, jsou hodnoty z předminulého snímku uměle nulovány, a jako bod v předminulém snímku je považován bod, který byl nalezen na konci smyčky, za předpokladu že v minulém snímku byl ztracen bod, a byl nalezen jiný vhodný kandidát. Pokud byl

v minulém snímku tento kandidát nalezen, byl uložen do proměnné *point_overview_full* a v aktuálním snímku je tedy použit jako výchozí bod, protože cost funkce potřebuje minimálně 2 nalezené body, aby mohla být sestrojena trajektorie. Do proměnné *point_overview_help* jsou tedy ukládány hodnoty před nalezením nových bodů, abychom věděli, že došlo ke ztrátě bodu a z této proměnné se také získávají body z minulého snímku (abychom mohli snáze detekovat ztrátu).

Po získání veškerých bodů je provedeno hledání bodů v aktuálním snímku s nastavenými hodnotami prahu. Po nalezení veškerých bodů z aktuálního snímku je použita funkce *FindallCandidateLoop*, která má za úkol nalézt body, které se vyskytují v oblasti bodů z minulého snímku vymezené poloměrem (*Vmax*). Funkce tedy již nehledá kombinaci všech možných bodů, ale počítá s minulými a předminulými body jako s pevnými hodnotami a hledá kandidáty pouze v aktuálním snímku a vyřazuje body, které nejsou uvnitř vymezené oblasti. Následně je použita funkce *PointsumLoop*, která seskupuje všechny nalezené kandidáty do jedné proměnné. Dále je opět ve funkci for cyklus pro umělé navýšení ztracených bodů. A následně je použita funkce *findbestcandidateloop,* která opět počítá hodnotu cost funkce pro danou trajektorii, ovšem jelikož je s předchozími hodnotami nakládáno jako s konstantami, hledá pouze kombinace s aktuálně nalezenými body a na základě nejmenší „ceny“ trajektorie je opět vybrán nejvhodnější bod a ostatní body, u kterých nebyla trajektorie nalezena, jsou ztraceny (*B_body*).

Pokud byl v aktuálním snímku některý bod z dané oblasti ztracen, nebo nebyl ani nalezen, přichází na řadu druhý detektor, který hledá významné body v celém obrazu. Po jejich nalezení je na tyto body aplikována funkce *FindBestPooint*, která je stejná jako funkce *findbestpoint3* a má za úkol nalézt nejsilnější body a okolí, do kterého nezasahují žádné již nalezené body. Poté je na nalezené body v aktuálním snímku, a na body ztracené v předchozích snímcích (*B_body*) aplikována funkce *ChooseBestRow*, která se snaží nalézt body, které mají souřadnice blízké bodům ztraceným (protože kvůli MB mohlo dojít ke ztrátě bodu pouze v 1 snímku, ale ve druhém již mohl být nalezen) a pokud tyto body nalezne, snaží se přeskládat nalezené body v trajektorii tak, aby seřadila ztracené a nalezené body do stejných řádků (kvůli lepší přehlednosti), aniž by zasáhla do správných trajektorií. Jako vstupní parametr vstupuje proměnná *B_body* z předchozího snímku, ovšem pokud je v této proměnné souřadnice ztraceného bodu, znamená to, že posední známá pozice tohoto bodu byla v předešlém snímku (čili porovnáváme aktuální a předminulé pozice).

V tomto okamžiku (po nalezení správných řádků) již v hlavní smyčce proběhne přiřazení nových bodů na místo ztracených do proměnné *point_overview_full* a také do proměnné *F_body*, ve které se nacházejí nalezené a použité body v aktuálním obraze. Poté proběhne zobrazení trasovaných bodů a hlavní smyčka pokračuje v další iteraci. Po skončení hlavní smyčky následuje proces *postprocessing*, který se snaží spojit ztracené trajektorie a doplnit chybějící body v této chybějící trajektorii (doplnění bodů probíhá s malou hodnotou prahu detektoru v definovaném malém okolí mezi ztraceným a nalezeným bodem). Více bude tento proces popsán níže.

Jako první je volána funkce *Trajectory2*, která z dostupných nalezených bodů a poloměru, které jsou zároveň vstupní funkce, rozloží skupiny bodů z daných oblastí na jednotlivé trajektorie. Při každé ztrátě bodu se tedy s dalším nalezeným bodem v dané oblasti začne tvořit nová trajektorie, do doby než jsou vytvořeny všechny trajektorie. Na začátku této funkce se nejprve uloží do proměnné *Xchange* čísla snímků, ve kterých dochází ke ztrátě trajektorie, zároveň se do proměnné *last_trajectory* uloží hodnota posledního snímku, na kterém byla nalezena trajektorie a do proměnné *last_poss* se uloží počet trajektorií v daném řádku (ten je dán délkou proměnné *Xchange*). Dále se ze zjištěného počtu počtu trajektorií počítá délka jednotlivých trajektorií, které jsou uloženy v proměnné *Xdiff*. Dále se poté vytvoří trojrozměrná matice, do které se ukládají za sebe jednotlivé trajektorie pro daný řádek (danou oblast)do proměnné *alltraject*. Nakonec je vytvořena z této matice dvourozměrná matice, která obsahuje veškeré trajektorie, které jsou poskládány za sebe a lze si tak prohlédnout délku (počet snímků) a také počet trajektorií.

Dále následuje funkce *CompleteTraject*, která se snaží z výstupu předchozí funkce nalézt trajektorie, které náleží stejnému trasovanému bodu a spojí je dohromady. Hledání těchto trajektorií probíhá od konce jedné trajektorie a vyhledává se v následujících 1:*max* snímcích, zda nebude nalezen bod, který svojí vzdáleností ještě náleží do dané oblasti. Tím že je jeho vzdálenost blízká v definovaném okolí znamená, že by se mělo jednat o stejný trasovaný bod, který mohl být kvůli MB na chvíli ztracen. Je-li takový bod nalezen, je zařazen za aktuální zkoumanou trajektorii na příslušné číslo snímku. V trajektorii tedy vznikne díra, která může být v rozmezí 1:*max-1* snímku. Na konci funkce se ještě zkoumá, zda nedošlo k detekci stejných bodů v jednom snímku, což by signalizovalo, že tento bod může náležet více trajektoriím, což by se stát nemělo.

Další funkcí, která následuje je funkce *FillHole*, která se snaží vyplnit díry v jednotlivých trajektoriích, jelikož každá trajektorie má svůj vlastní sloupec a obsahuje pouze jednu jedinou trajektorii, tak tato funkce se snaží nalézt díry v jednotlivých trajektoriích. Tyto díry v trajektoriích signalizují, že v nich byl ztracen bod a za několik snímků opět nalezen jak bylo řečeno výše. Funkce začíná detekcí počtu děr a místa, ve kterém se díra nachází (snímek, číslo trajektorie) ve for cyklu pro jednotlivé sloupce. Poté následuje další for cyklus, který se pokouší nahradit detekované díry pomocí posledního známého bodu, nebo prvního nalezeného (pokud je více děr v jedné trajektorii, najde nejbližší poslední/začínají bod konkrétní trajektorie). Pro tento bod se aplikuje již známá funkce *FindROI*, která vymezuje oblast hledání nového bodu a je vypočítána pomocí zadaného poloměru rychlosti. V této oblasti je hledán bod bez jakýchkoliv prahů. Pokud je nalezeno více významných bodů, můžeme v této oblasti nalézt nejvhodnější bod buď podle vzdálenosti od posledního známého bodu (*method*=1), nebo nalézt nejsilnější bod z nalezených (*method*=0). Pokud není nalezen žádný bod, funkce pokračuje k další díře až po poslední trajektorii.

Následuje funkce pro spojení vytvořených trajektorií zpět do původní formy. K tomu je využívána funkce *MergeTraject* kvůli tomu, že jsme však zaplnili díry, získali jsme v některých snímcích více bodů, než zadaný maximální počet, jelikož se při ztrátě automaticky hledaly další body. Tím dochází k tomu, že v některých místech máme více trajektorií, než zadaný počet trajektorie, které jsou navíc, mají sice většinu svého místa prázdné, nicméně jsou stále součástí skupiny trajektorií a tato funkce se snaží poskládat všechny trajektorie do minimálního počtu řádků. Začátek funkce začíná detekcí hlavní a vedlejší trajektorie, kterou zprostředkovává funkce *MainOtherTrajectoryDetect*. Tato funkce nalezne *MaxNrTrackedPts* hlavních trajektorií (ty které začínají nejnižšími snímky) a poté, jakmile jsou nalezeny hlavní trajektorie je pomocí proměnné *other*=1 hledána vedlejší trajektorie (opět nejnižší číslo snímku) toto hledání trajektorií je neustále aktualizováno, jelikož se trajektorie v průběhu funkce neustále skládají a minimalizují, proto je nutné hledat hlavní i vedlejší trajektorie při každé iteraci. Po nalezení hlavních trajektorií se naleznou počátky a konce jednotlivých trajektorií v aktuální iteraci pomocí funkce *startendtrajectory*, která vrací počáteční a koncové hodnoty snímku těchto trajektorií. Při získání konce jedné trajektorie se hledá, který začátek trajektorie je nejblíže a celá tato nejbližší trajektorie je přesunuta za aktuální trajektorii, přičemž místo přesunuté trajektorie je nulováno a vymazáno (redukce). To je také důvod neustálého hledání nového začátku a konce trajektorií. Celý tento proces se opakuje do doby, než jsou všechny trajektorie v minimálním počtu.

Nakonec je použita funkce *resizepoint*, která přeskládá trajektorie zpět do stavu, který obsahuje pro konkrétní snímek veškeré nalezené body (těch ovšem může být více, než maximální počet). Tato funkce přeskládá body do takové proměnné, která poté umožňuje následné porovnání s referenčními body. Po skončení této funkce tedy program porovnává nalezené body stejně jako v ostatních použitých metodách.

4.4.3 Implementace metody RC (rank/subspace constraint)

Poslední implementovanou metodou je metoda zamezení na omezení geometrického pohybu, pomocí penalizačního výrazu. Metoda je z uvedených nejnovější a je založena na principu pohybu bodů jako skupiny. Základní proces *inicializace* je stejný, jako bylo uvedeno při celkovém popisu metod. Rozdíl začíná až uvnitř hlavní smyčky, kde je začíná vytváření proměnných předchozího (*Iold*) a současného (*Inew*) snímku a také průběžným vytvářením *history_matrix* v tvaru požadovaném pro posuvné okno uvedené v popisu metody $2(L+1)xF$. Poté následuje definice celkem 4 proměnných, které rozhodují o výsledném výpočtu penalizačního výrazu P a také o způsobu vyhodnocení celkové energie, která však nebyla v článku, ale byla přidána pro ověření, jak pracuje metoda, pokud sice počítáme celkovou energii, ale posouváme každým bodem zvlášť tak, aby se co nejvíce shodoval s šablonou. Po definici všech proměnných, které jsou zároveň vstupem funkce, se vypočítá celková energie srovnání šablon pomocí funkce *Evaluate_Energy*. Na začátku této funkce se pomocí zadaných parametrů zvolí hodnota normalizačního koeficientu *m*, který uvádějí autoři v doplňujících materiálech a také je vypočten koeficient α. V následující části jsou vytvořeny pevné šablony, které jsou získány z pozice významného bodu v minulém snímku s pevně definovanou velikostí, která je parametrem funkce. Vytváření šablony probíhá 2 způsoby. Prvním způsobem je použití funkce *genMesh*, která na základě zadaných hodnot rozměru šablony vygeneruje potřebné souřadnice, od kterých lze vyříznout šablonu z obrazu, takto vyříznutá šablona je poté interpolována, pokud nemá celočíselné souřadnice. Druhý způsob nastane tehdy, pokud vygenerované souřadnice přesahují přes okraj obrazu. V tomto případě je použita funkce *findROIoldTemplate*, která ošetřuje překročené hodnoty šablony tím, že vytvoří stejně velkou šablonu a hodnoty, které přesahují přes obraz, jsou naplněny hodnotami *NaN*. Dále následuje detektor významných bodů, který slouží k odstranění šablon, které se kvůli skupinovému pohybu dostaly do míst, na kterých by neměly být (např. stěna, rovná plocha skříně), protože se v okolí nenachází žádný významný bod. Odstranění probíhá tak, že se detektor s velmi nízkým prahem pokusí nalézt v šabloně významný bod a pokud nebude žádný

nalezen, předpokládá se, že šablona je mimo významný bod. Toto odstranění je v této funkci také navíc. Na závěr jsou vypočítány hodnoty startovních souřadnic *startstred2*, od kterých se bude pohybovat šablona při hledání optimální polohy, která nejvíce odpovídá šabloně z předchozího snímku. Toto vyřezání šablon je prováděno ve for cyklu. Po jeho skončení následují další 2 for cykly, které mají za úkol posouvat šablonami v X a Y ose (šířka, výška). Pro každý posun je vypočítán aktuální společný stav/pozice bodů *joint_state*.

Uvnitř této smyčky je použita funkce pro vyčíslení penalizačního výrazu *Evaluate_P*. Funkce po získání dostatečného množství snímků (L) vytvoří matici, která se skládá z L minulých hodnot a přidání k této pozici ještě aktuální pozice, které jsou proměnné v závislosti na posunu. Vytvoříme tedy matici M o rozměrech $2(L+1)xF$. Pokud v průběhu L snímků došlo ke ztrátě nějaké bodu, je celá trajektorie z L prvků vyřazena z této matice. Pokud je povoleno centrování matice (*center*=1) je od každé hodnoty X, Y v jednotlivých sloupcích odčítána jejich průměrná hodnota, pokud je centrování zakázáno, hodnoty zůstávají. Poté je počítán rozklad singulárních hodnot této matice (SVD) a podle zadaných parametrů (center, method) je vypočítána hodnota penalizačního výrazu P podle vzorců uvedených v teoretickém rozboru.

Po skončení výpočtu penalizační funkce následuje vyřezávání šablon pro aktuální pozici jednotlivých nalezených bodů pomocí funkce *findROIactualTemplate*, která je velmi podobná přechozí funkci na vyřezávání šablon liší se pouze ve způsobu generování rozsahu a umožňuje generovat rozsah pro body, které překročily okraj obrazu i pro body, které obraz nepřekročily. Poté je spočítána obrazová chyba (chyba šablon) z dvojic vyřezaných šablon. Pokud k překročení došlo, program vypočítá počet hodnot, které překročily obraz a normalizuje chybu šablon tak, že podělí výslednou hodnotu chyby procentem neztracených bodů, aby nedošlo k tomu, že obraz, který bude mít většinu bodů ořezaných, byl vyhodnocen jako nejvhodnější. Každá hodnota této chyby je potom ukládána do vektoru podle příslušného posunutí. Poté se podle volby proměnné *total* určí, zda se bude počítat globální funkce energie (*total*=1), která používá pouze 1 vektor, kde se veškeré chyby na příslušných pozicích sčítají, nebo se bude počítat obrazová chyba pro každý obraz zvlášť (*total=0*). Poté jsou hodnoty energií násobeny koeficientem α zvoleným v parametrech a také se k tomuto výrazu přičte proměnná P. Poté je pomocí proměnné *total* také rozhodnuto o způsobu posouvání bodů, zda se bude s body posouvat jako v celkem *total=1*, nebo s každým bodem zvlášť *total=0*. Tím končí funkce *Evaluate_Energy*. Zbytek programu již obsahuje pouze hledání nových bodů pomocí známé funkce *findbestpoint3*, který je umístěn v hlavní smyčce.

4.5 Popis vlivu jednotlivých parametrů metod na trasování

Tento oddíl je zaměřen na zkoumání nastavení parametrů jednotlivých metod a jejich dopad na trasování. Každá změna některého parametru totiž může trasování ovlivnit příznivě i nepříznivě a tento vliv bude popsán pro každou metodu zvlášť s cílem nalézt nejlepší nastavení pro jednotlivé metody.

První parametr, který sice neovlivňuje schopnost algoritmů trasovat, ale je přítomný v každém algoritmu, je proměnná *MaxNrTrackedPTs*, která určuje maximální počet bodů, které se budou trasovat. Tato hodnota byla u pořízených videosekvencí volena nejčastěji kolem hodnoty 10, lze však trasovat mnohem více bodů. S touto hodnotu je také částečně spojena proměnná *neighborhood*, která určuje okolí, ve kterém dochází k potlačení významných bodů tak, aby každý významný bod byl v tomto okolí jediný. Toto potlačované okolí je voleno hlavně z důvodu, abychom při trasování a možném driftu trasovaných hodnot nezačaly trasovat 2 stejné body. Tento parametr je také spjat s hodnotou počtu trasovaných bodů, obecně platí, že čím více bodů chceme trasovat, tím menší by měla být hodnota tohoto okolí, zůstane-li hodnota okolí velká a budeme chtít trasovat velký počet bodů, dojdu k tomu, že ani nebudeme schopni v žádném snímku dosáhnout maximálního počtu bodů. S hodnotou okolí je také spojena hodnota *radius*, která má poloviční hodnotu a určují poloměr tohoto okolí. Dalším důležitým parametrem při trasování je hodnota *threshold*, která určuje minimální práh, který musí splnit „síla“ významného bodu, aby byl považován za nalezený. Tato hodnota je většinou nastavená pro inicializaci na vyšší hodnotu, abychom již od začátku inicializace sledovali kvalitní body. Tato hodnota je většinou volena kolem hodnoty 1000. Dalším parametrem je *threshold2*, která opět určuje práh detektoru významných bodů, tentokrát však pro trasování v hlavní smyčce, a proto je většinou tento parametr nastavován na nižší hodnotu, aby byl schopen nalézt více bodů, ze kterých si již algoritmy zvolí nejvhodnější pro své použití, protože společně s body lze získat i jejich „sílu“. Tato hodnota je obvykle volena kolem 0.1 pro detekci i velmi slabých bodů, případně hran a poté kolem hodnoty 0.6 a výše pro detekci kvalitnějších bodů. Volba tohoto parametru závisí na konkrétním algoritmu.

4.5.1 Volitelné parametry metody KLT

Tato implementace obsahuje několik volitelných parametrů, které ovlivňují trasování. Mezi základní patří již zmiňovaná volba prahu detektoru *threshold2*, která byla zkoumána pro celkem 4 hodnoty vzdálené od sebe po násobcích 10, nebo 5 tak, aby byla viditelná změna tohoto parametru. Při nastavení tohoto parametru na velmi nízkou hodnotu (0.001) je trasování úspěšné jen pro body nalezené při inicializaci s vysokým parametrem prahu prvního detektoru, v momentě kdy jsou tyto silné body ztraceny a hledají se nové, jsou většinou nalezeny body, které jsou k trasování nevhodné (stěna, plochy na kterých nelze nějaký bod zachytit) a trasování v tomto případě i s velmi přesnými parametry nedokáže udržet bod, jelikož jeho trasování je velmi obtížné. Pro vyšší parametr (0.01) je trasování bodů již výrazně lepší, nejen po inicializaci, ale i v průběhu trasování je občas nalezen některý silný bod, který se dá úspěšně trasovat. Nicméně stále dochází k nalezení bodů, jejichž významnost a kvalita je velmi slabá, nalezené jsou už ale například hrany, což je v jistých ohledech lepší, než stěna. Pro vyšší parametr (0.1) již trasování probíhá velmi dobře, jsou trasovány body, které vykazují vysokou kvalitu a v průběhu trasování většinou nejsou ztraceny a detekce nevýznamných bodů (stěny) již nenastane, stále je ale občas nalezena silná hrana. U zvolení ještě vyššího parametru (0.5) z rozsahu <0,1> probíhá již trasování na velmi vysoké úrovni i pro vysoký počet bodů. I když pro vysoký počet bodů dochází k detekci hran, algoritmus je schopen i tyto hrany udržet a správně trasovat. Toto nastavení tohoto parametru obecně platí pro všechny 3 algoritmy a u ostatních již nebude popisován.

Dalším významným parametrem je *NumPyramidLevels*, který určuje, zda bude trasování probíhat pyramidálně, a kolik pyramid bude použito pro toto trasování, hodnoty této proměnné by měly být voleny v rozsahu <1-4>, více pyramid je již zbytečné používat, protože tento počet pyramid je vyhovující. Větší počet pyramid je schopen nalézt a zachytit probíhající pohyb ve velkém měřítku (velkých vzdálenostech od poslední známé hodnoty). Pokud není použita pyramidální reprezentace (*pyramid*=1), může dojít k tomu, že pokud ve videosekvenci prudce pohneme kamerou, nebude algoritmus schopen kvůli vysoké rychlosti pohybu, zachytit trasovaný bod, protože jeho vzdálenost bude příliš velká. Při použití alespoň jedné a více pyramid však již jsme schopni takovýto pohyb nalézt, jelikož s každou pyramidou se zvyšuje rozsah pro vyhledání obrazové šablony. Pozorování změny je nejvhodnější buď pro žádnou pyramidu, nebo např. pro 3 pyramidy. S použitím pyramidální interpretace jsme tedy schopni zachytit větší vzdálenosti, ovšem za cenu zvýšení výpočetního výkonu. V experimentu je počet pyramid 3, pro nejlepší zachycení pohybu.

Následuje parametr *MaxBidirectionalError*, který určuje hodnotu prahu chyby při dopředném-zpětném trasování. Tento druh parametru spočívá v tom, že se nejprve standartně trasuje bod z přechozího snímku na aktuální, ovšem poté se stejný nalezený bod trasuje z aktuálního obrazu zpět. Hodnota rozdílu počtu pixelů těchto dvou trasovaných bodů musí být menší, než zadané číslo (vhodné hodnoty jsou mezi 0 až 3 pixely, přičemž při nastavení *Inf* tento parametry není počítán. Použití tohoto parametru je účinný způsob jak odstranit body, které nejsou vhodné pro trasování. Tato hodnota je nastavena na *Inf* při testování.

Následující volitelný parametr je *BlockSize*, jenž určuje rozměry okolí, kolem kterého budou body trasovány (jinými slovy rozměry šablony). Tyto hodnoty musí být liché, aby měli centrální pixel. S vysokou hodnotou rozměrů šablony však roste výpočetní náročnost. Pokud je však tato hodnota nastavena na nejnižší možnou [5,5] dochází při trasování k tzv. driftu bodů, zejména při detekci hran. Protože má šablona male rozměry a většina hran je v jistém okolí velmi podobná, dochází k tomu, že trasovaný bod různě putuje po okolí této hrany. V ideálním případě je dobré volit rozměry této šablony větší, kolem hodnoty [21,21] kdy je již okolí šablony tak velké, že dochází k úspěšnému trasování, zejména u slabších bodů (hrany), jelikož je počítaný rozdíl mezi špatnými šablonami větší. Při experimentech je hodnota nastavena na [31 31].

Poslední nastavitelným parametrem je *MaxIterations*, který určuje maximální počet iterací, kterými je omezena metoda pro nalezení nejoptimálnější polohy trasovaného bodu v aktuálním snímku. Algoritmus většinou konverguje v okolí hodnoty 10, při nastavení hodnoty na nízkou úroveň může dojít k tomu, že pozice nalezeného bodu v aktuálním obraze nebude úplně přesná, ale jen přibližná. Při nastavení vyšší hodnoty jsou poté souřadnice nalezeny s nejvyšší možnou přesností, za předpokladu, že daný bod lze nalézt. Tato hodnota je nastavena v našich testovacích podmínkách na 30.

4.5.2 Volitelné parametry metody IPAN

Jedním z nejdůležitějších parametrů této metody je *Vmax*, který určuje rozsah vyhledávací oblasti významného bodu. Jedná se o odhad rychlosti pohybu v obrazových souřadnicích (rychlost představuje počet pixelů), který vymezuje hledaný okruh. Parametr je podobný použití pyramidální implementace, pokud je jeho hodnota nastavena na vyšší hodnotu (také je schopen zachytit rychlý pohyb). Tento parametr nám totiž v podstatě říká, jak velká má být hodnota oblasti, ve které vyhledáváme trasovaný bod. Čím je tato oblast větší, tím větší je šance, že nalezneme tento bod při prudkém, nebo zvýšeném pohybu. Ovšem s touto vysokou hodnotou je také spojena výpočetní náročnost. Tato proměnná je použita také při

trasování pomocí třetí metody (RC tracking). Nastavení těchto hodnot se může lišit při *inicializaci*, kdy pro co nejpřesnější počátek trajektorie volíme hledací oblast vyšší, a v průběhu *hlavní smyčky*, kde bývá hodnota většinou volena nižší, z důvodu snížení výpočetní náročnosti. Při inicializaci je tato hodnota nastavena jako *Vmax*=20. V průběhu hlavní smyčky má volba velikosti této proměnné zásadní vliv na trasování. Pokud je hodnota nastavena jako $Vmax \leq 5$ je kvalita trasování vyhovující pouze, pokud je pohyb s kamerou velmi pomalý a nedochází ani k pohybovému zkreslení, v momentě kdy však pohyb začne být rychlejší, nedokáže algoritmus již zachytit hledaný bod, ani v případě, že je dobře trasovatelný, protože nepředpokládá takovou prudkou změnu souřadnic a spousta bodů v průběhu tohoto prudkého pohybu se ztracena. I když jsou nalezeny nové body, jsou většinou také okamžitě ztraceny, dokud se pohyb trochu neustálí, pro tyto parametry tedy trasování nefunguje příliš dobře. Při volbě $Vmax > 5$ a zároveň $Vmax < 16$ se pro testované videosekvence ukázalo, že tento rozsah parametrů je nevhodnější z hlediska kompromisu mezi výpočetní náročností a kvalitě trasování. Pro tyto hodnoty je totiž trasování již na úrovni, že pokud nedochází k velkému vlivu MB (který řeší postprocessing), tak je algoritmus schopen nalézt většinu významných bodů i při prudkých pohybech. Přičemž je výpočetní náročnost stále ještě ve vyhovujících mezích. Pro nastavení tohoto parametru $Vmax \geq 16$ již doba výpočtu při jednotlivých výpočtech stoupá, a počet zachycených bodů při prudkých pohybech není o mnohem vyšší. Proto je při experimentech tento parametr nastaven na hodnotu *Vmax=15*.

Dalším nastavitelným parametrem je *MaxCandidates*, který určí maximum počtu kandidátů vhodných pro sestrojení trajektorie. Minimální hodnota pro tento parametr je 1. Při experimentech však většinou stačilo mít tuto hodnotu nastavenou na 2-3 kandidáty, jelikož tento počet byl většinou dostačující a více kandidátů na danou trajektorii se většinou ani nepodařilo získat, protože většina bodů se opakovala. Hodnota tedy byla nastavena jako *MaxCandidates=3*.

Významnými parametry v tomto algoritmu jsou také parametry *threshold2, threshold3*, které opět určují práh pro detektor významných bodů. Opět je zde jiná hodnota pro *inicializaci*, kde je potřeba detekovat co možná nejsilnější body a proto je nastaven parametr *threshold2=0.6* pro detekci kvalitnějších bodů. Pokud se ovšem program nachází v *hlavní smyčce,* jsou parametry těchto dvou proměnných měněny. Tyto parametry jsou odlišné z toho důvodu, že metoda IPAN nijak nehledí na kvalitu významných bodů, ale penalizuje pouze změny ve směru a velikosti bodu. Z tohoto důvodu je vhodné nastavovat parametr *threshold2* na co nejnižší úroveň tak, aby bylo nalezeno co nejvíce možných bodů, ze kterých si poté cost funkce zvolí nejvhodnější. Při nastavení parametrů na příliš nízkou

hodnotu (kolem 0.02) se však většinou stává, že jsou zvoleny nevhodné body a trasování driftuje, nebo se začne trasovat špatný bod, který je velmi blízko předchozímu. Naopak při nastavení vyšších hodnot (kolem 0.2) dochází k tomu, že cost funkce nemá moc adeptů, protože se kvůli MB nemusí podařit najít dostatečně silný bod pro nalezení optimálního a to může vést také ke špatnému trasování. Proto byla zvolena hodnota pro testování *threshold2*=0.1, tak aby byl zajištěn dostatečný počet kandidátů a zároveň jejich jistá kvalita. Naopak pro detekci *nových* bodů, které zaplní místo po ztracených bodech je vhodné mít parametr pro detektor *threshold3* nastaven na vyšší úroveň tak, abychom vytvořili alespoň pro začátek trasované trajektorie silný bod. Pro tento parametr byla zvolena hodnota *threshold3=0.4*, která zajišťuje jistou spolehlivost bodů při trasování a zároveň lze při vyšším počtu bodů nalézt dostatečný počet kandidátů pro vytvoření nové trajektorie.

Posledním a také velmi důležitým parametrem této metody je nastavení vah pro cost funkci *w1 a w2*, které jsou spolu spojeny tak, že kvůli normalizaci by měli dohromady dávat hodnotu 1. Proto budeme popisovat pouze nastavení parametru váhy *w1* a parametr *w2* bude dopočet do hodnoty 1 od této hodnoty. Parametr *w1* penalizuje změnu směru bodu, parametr *w2* naopak rychlost pohybu bodu. Pokud je parametr *w1* nastaven na příliš malou hodnotu (0.1,0.2) dochází při trasování často ke ztrátě trasovaného bodu, a musí se nalézt jiný bod, to je způsobeno hlavně při vyšších rychlostech, na které je kladen při tomto nastavení důraz. Při nastavení tohoto parametru na příliš vysokou hodnotu (0.8,0.9) naopak dáváme přednost nejmenší změně ve směru trajektorie, což sice nevede většinou ke ztrátě bodu, ale k situaci že trasovaný bod se snaží udržet svoji polohu, čímž dochází v mnoha případech k driftování bodu. Proto byla tato hodnota při hledání nejoptimálnějších parametrů pro videosekvence nastavena na hodnotu *w1*=0.4.

4.5.3 Volitelné parametry metody RC (rank constrained)

Hlavní myšlenkou této metody je předpoklad, že pohyb bodů bude skupinový a body se budou pohybovat stejně, pokud se bude předpokládat, že ve scéně se nenachází pohybující se tělesa (α_{strong}=1). Pokud se bude předpokládat, že se tělesa mohou pohybovat různým směrem, je vhodné volit parametr (α_{strong}=0), při našich definovaných podmínkách však scéna neobsahuje žádné pohybující se tělesa a pohybuje se pouze kamera se statickou scénou. Proto bude tento parametr testován pouze pro hodnotu (α_{strong}=1).

Dalším velmi důležitým parametrem je opět nastavení prahu detektoru *threshold2, threshold3*. Třetí práh určuje minimální hodnotu kvalitu nově nalezených bodů, pokud dojde ke ztrátě některého bodu, z tohoto důvodu je nastaven na vyšší hodnotu tak, aby nalezl kvalitní body, které budeme trasovat, jelikož tato metoda potřebuje alespoň pár silných bodů, aby dokázala správně pohybovat body. Proto je tato hodnota nastavena jako *threshold3=0.3*.

Druhým parametrem je *threshold2*, který v tomto případě slouží k vyřazení bodů, které se z důvodu skupinového posuvu dostaly mimo svoji oblast (například se posunuly na stěnu) a negativně ovlivňují výsledky trasování, protože nemají jednu nejvhodnější polohu, ale většinou takovému bodu vyhovuje spousta posuvů. Tento způsob vyřazení je provizorní, jelikož autoři článku neuvedli metodu, kterou odstraňují špatně trasované body. Hodnoty tohoto parametru jsou však velmi proměnné, pro některé videosekvence stačí nízké hodnoty (0.05-0.1) a pro některé videosekvence, které vykazují značný MB a velmi prudké pohyby kamerou, dosahují hodnoty až (0.9) a stále nejsou veškeré nevhodné body vyřazeny. Jelikož metoda potřebuje alespoň pár sledovatelných bodů k správné funkci, je v těchto případech trasování velmi nekvalitní.

Další parametr je hodnota délky historie *L*, která určuje velikost posuvného okna přes poslední historii *L* snímků. Na tomto základě se počítá penalizační výraz *P*. Autoři ve svém článku uvádí doporučené hodnoty v rozsahu 5 až 10. Pro hodnotu 10 se však předpokládá, že v průběhu 10 bodů nebyl bod ztracen, a tak byla zvolena kvůli různorodým videosekvencím hodnota *L*=5. Nastavení tohoto parametru ovlivňuje výpočet výrazu P, protože algoritmus pomocí této délky historie odhaduje a omezuje geometrický pohyb a také pomocí této hodnoty odvozuje model kamery pomocí výpočtů uvedených v článku.

Následující parametr je *center*, jež slouží jako proměnná, která říká, zda se bude matice historie centrovat (*center=1)*, nebo nikoliv (*center=0)*. Centrování matice pomáhá při odhadu modelu kamery, jelikož transformuje afinní omezení pohybu na lineární a pomáhá tak lépe určit hodnotu výrazu P. Jelikož je proto lepší použít centrování z důvodu větší přesnosti bude tento parametr nastaven při testování na hodnotu (*center=1).*

Method je další nastavitelný parametr, který určuje metodu, kterou používají autoři článku *method=0* je zvolení explicitní faktorizace. Jde o způsob vyhodnocení penalizačního výrazu P, kdy je penalizovaná energie která se nachází mimo předpokládanou dimenzi (předpoklad dimenze je závislý na centrování). Tato energie je nižší, než uvnitř této dimenze, protože je zde předpoklad, že body „žijí“ uvnitř této dimenze a energie mimo tuto dimenzi je spíše odchylka. Nižší

energie je však zohledněna parametrem *m*. Při volbě této metody Při zvolení *method=1* je vybrán výpočet penalizačního výrazu pomocí nukleární normy, která penalizuje celkovou energii, kterou body a model kamery vykazují. K výpočtu je použit součet singulárních hodnot SVD. Pro testování bude použito *method=1*, protože obě metody vykazují malé rozdíly výsledků.

Parametr, který je shodný s parametrem popsaným u metody IPAN, označovaný jako *Vmax* znamená opět předpoklad o maximální obrazové rychlosti, jakou se může bod pohybovat. Jelikož vytvořená funkce neobsahuje pyramidální interpretaci, je vhodné volit tuto hodnotu vyšší (kolem 30), aby bylo možno zachytit i prudké pohyby a tím získat co nejpřesnější šablony, které budou porovnávány. Pokud bude tato hodnota malá, bude docházet ke ztrátě bodů, nebo špatné polohy významných bodů, což by vedlo i ke špatnému vytvoření šablony a tím pádem by byl bod špatně trasovaný. Vysoká hodnota však na druhou stranu velmi výrazně zvyšuje výpočetní výkon. Proto je tato hodnota v algoritmu volena jako *Vmax=30*.

Poslední parametr, který také výrazně ovlivňuje jenom kvalitu, ale i výpočetní rychlost algoritmu je *tempsize*, jedná se o hodnotu velikosti šablony, která je vyřezávána pro předchozí i aktuální body. Hodnoty této proměnné by měly být liché, aby šablona měla středový pixel. Pokud budou hodnoty voleny malé (*tempsize≤9)*, může docházet k tomu, že rozdíly v obrazové chybě budou velmi nízké i pro nesprávnou pozici šablony, čímž může při takové špatné detekci vést k nesprávnému vyhodnocení pohybu, a body mohou být posunuty na nesprávná místa, což by mělo za výsledek špatné trasování. Při zvolení vysoké hodnoty (*tempsize≥21)* však dochází ke značnému nárůstu výpočetní doby. Proto je hodnota volena jako kompromis mezi těmito parametry na (*tempsize=15)*.

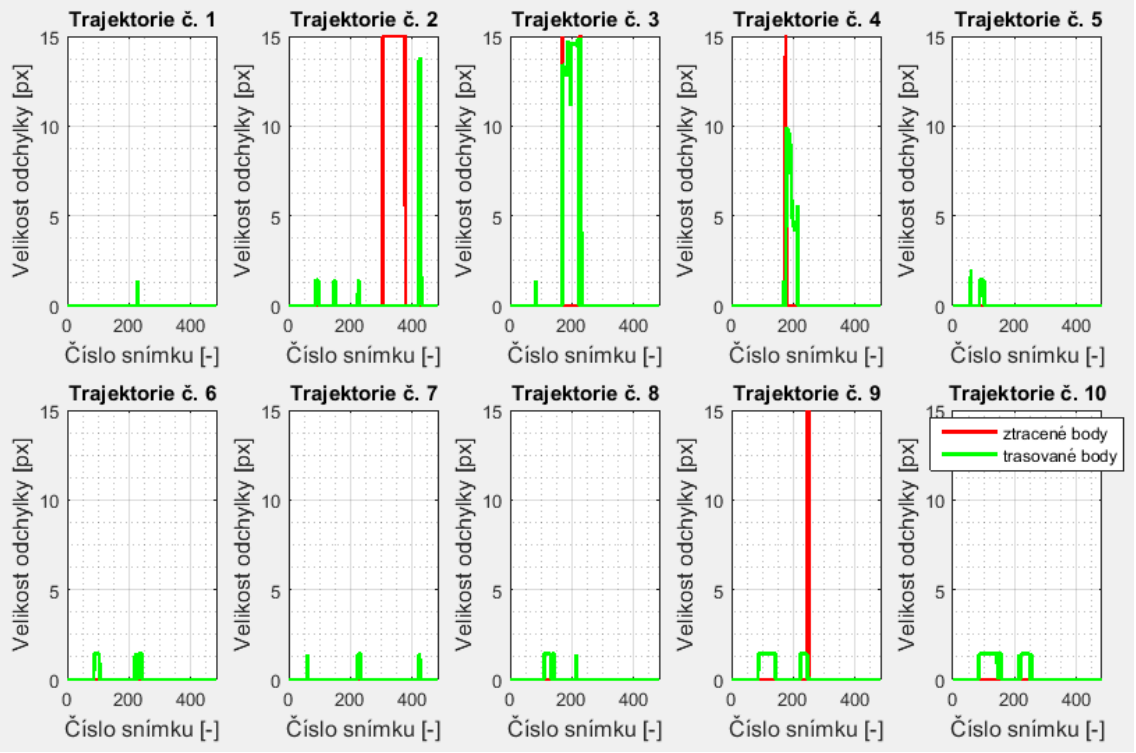
5 SROVNÁNÍ JEDNOTLIVÝCH METOD

Zdejší část je věnována porovnávání jednotlivých implementovaných algoritmů. Porovnávány budou veškeré vlastnosti a parametry, které jsou uvedeny v sekci *Návrh kritéria hodnocení*. Porovnávání bude probíhat podle daných kritérií postupně, pro všechny videa a zároveň pro všechny algoritmy s jejich nejlepšími nastavenými parametry. Aby bylo dosažené jisté normalizace, budou všechny důležité parametry nastaveny na stejnou hodnotu, aby byly zajištěny stejné podmínky pro všechny metody i videosekvence. Což znamená nastavení parametru *MaxNrTrackedPts* =10. A hodnota okolí *Neighborhood*=31 pro videa natáčené mobilem a *Neighborhood*=51 pro videa pořízené z fotoaparátu. Parametry prahů detektorů a ostatních hodnot budou nastaveny na hodnoty, při kterých trasování probíhá nejlépe, a tyto parametry jsou přednastaveny. Z důvodů nutných úprav programu pro správné testování byly vytvořeny 2 složky, které budou na CD k dispozici. První složka *samostatné vyhledávání* obsahuje zcela samostatné verze jednotlivých metod, které si sami hledají nové body při ztrátě některého trasovaného bodu. Druhá složka *s referenčními body* byla vytvořena záměrně kvůli testování. Složka obsahuje opět všechny implementované metody, ovšem s drobnými úpravami a to tak, aby při ztrátě některého trasovaného bodu metoda použila pro začátek trasování některý z bodů referenčních, který je ovšem v aktuálním snímku také nalezen (nelze vzít referenční bod, který již má trajektorii v průběhu, ale pouze ten bod, jehož trajektorie začíná na daném snímku). Těmito úpravami jsou zajištěny stejné podmínky pro všechny metody a zajištění toho, že budou porovnávány vždy jen referenční body a lze tak snadno pozorovat které metody trasují bod déle než by měli, případně kdy metoda ztratí bod dříve než by měl být ztracený, případně kdy dojde k trasování jiného bodu (např. kvůli překrytí). Všechny metody v obou složkách byly také na začátku doplněny možností výběru jednotlivých videí a s tímto výběrem jsou také přednastaveny hodnoty parametrů (lze změnit). Ve složce pro srovnání referenčních bodů je také na začátku trasování použita funkce *traject2* pro získání referenční trajektorie a také funkce *startendtrajectory* pro získání snímku, na kterém začínají a končí jednotlivé trajektorie (kvůli náhradě bodů referenčními). Dále obsahuje každý algoritmus smyčku pro odstranění bodů, jež nesplňují předpoklad potlačovaného okolí a překrývají se. Poté je do hlavní smyčky přidána smyčka, která zajišťuje právě hledání nových referenčních bodů, které mohou být použity, pokud dojde ke ztrátě trasovaného bodu. Byla také upravena inicializace tak, aby všechny metody měly stejné výchozí podmínky a to tím způsobem, že byly použity referenční body.

Pro tvorbu referenčních bodů byla použita metoda KLT s nastaveným maximálním počtem 30 trasovaných bodů. Poté byly ručně vyřazeny body, kde metoda selhávala (špatné osvětlení, detekce hran), nebo kde byly trasovány body, které měly být ztraceny (překrytí, změna vzhledu šablony), nebo nepřesné body (z důvodů pohybového zkreslení, kdy se body dostaly mimo trasovaný bod). Složka *tvorba refer*, ve které probíhalo manuální odstranění bodů je také obsahem CD, avšak tvoří pouze doplněk a s metodami trasování nesouvisí. Tato složka je obsažena uvnitř složky *samostatné vyhledávání*. Všechny algoritmy také obsahují testovací část, která je umístěna na konci každého programu. Zde je proveden výpočet odchylek od referenčních trajektorií, přehled, zobrazení grafů pro tyto odchylky (pro jednotlivé trajektorie, pro celkovou odchylku), přehrání videa spolu s trasovanými a referenčními body (které byly úspěšně trasované) a také výpočet dalších kritérií hodnocení algoritmů (počet trasovaných bodů atd.).

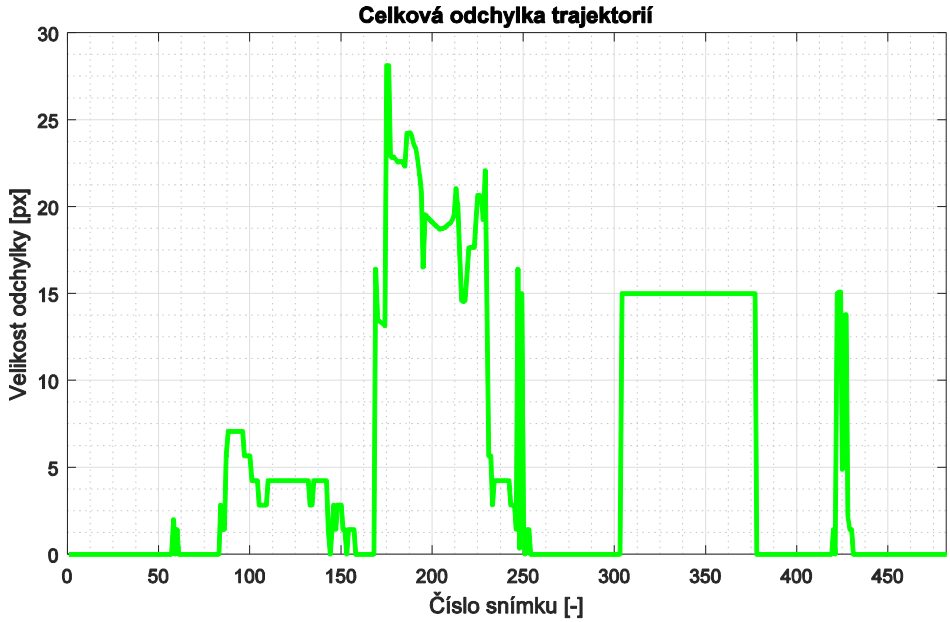
Pro určení odchylky od referenčních bodů byla použita funkce *DeviationOfRefer_last*, která po zadání referenčních, trasovaných bodů a poloměru vypočítá pro každý jednotlivý trasovaný bod odchylku od všech referenčních bodů (proměnná *dista2*) a body, které přesahují poloměr potlačovaných bodů, nebo jsou ztraceny (nulové) získají hodnotu *NaN* (*dista2*). Poté je pro každý jednotlivý bod vybrán referenční bod (pokud existuje), který má nejmenší odchylku od daného bodu a je uložen do proměnné *ClosesTraj* tato proměnná tedy obsahuje pouze body, které byly *trasovány*, ale zároveň byly shodné s referenčními, pokud se neshodovaly, byly nulovány. Na stejném principu je založena i proměnná *ClosesReferTraj*, která ale naopak obsahuje pouze *referenčními body*, které se shodovaly s trasovanými, zbytek je nulován. Dále jsou v této funkci je použita proměnná *deviationNaN*, která pro všechny body, kdy došlo ke ztrátě, nebo špatnému trasování, obsahuje maximální možnou odchylku od trajektorie. Dále proměnnou *deviation*, která obsahuje pouze odchylky jednotlivých bodů od jejich nalezených odpovídajících referenčních bodů (při ztrátě je hodnota rovna nule). Poslední proměnná *deviation_for_group*, obsahuje stejné hodnoty, jako proměnná *deviation*, ovšem hodnoty kdy došlo ke ztrátě, jsou nastaveny jako maximální možná odchylka (velikost poloměru). Tyto 2 proměnné slouží ke zhodnocení celkové odchylky pro jednotlivé snímky.

Dále v programu poté probíhá vykreslení odchylek jednotlivých trajektorií od referenčních bodů, jak lze vidět na Obrázek č. 13. Zeleně vyznačené jsou velikosti odchylek od referenčních bodů pro jednotlivý snímek. Červená barva signalizuje ztrátu bodu od referenční trajektorie v daném snímku.



Obrázek č. 13 Ukázka odchylek od referenčních bodů pro jednotlivé trajektorie

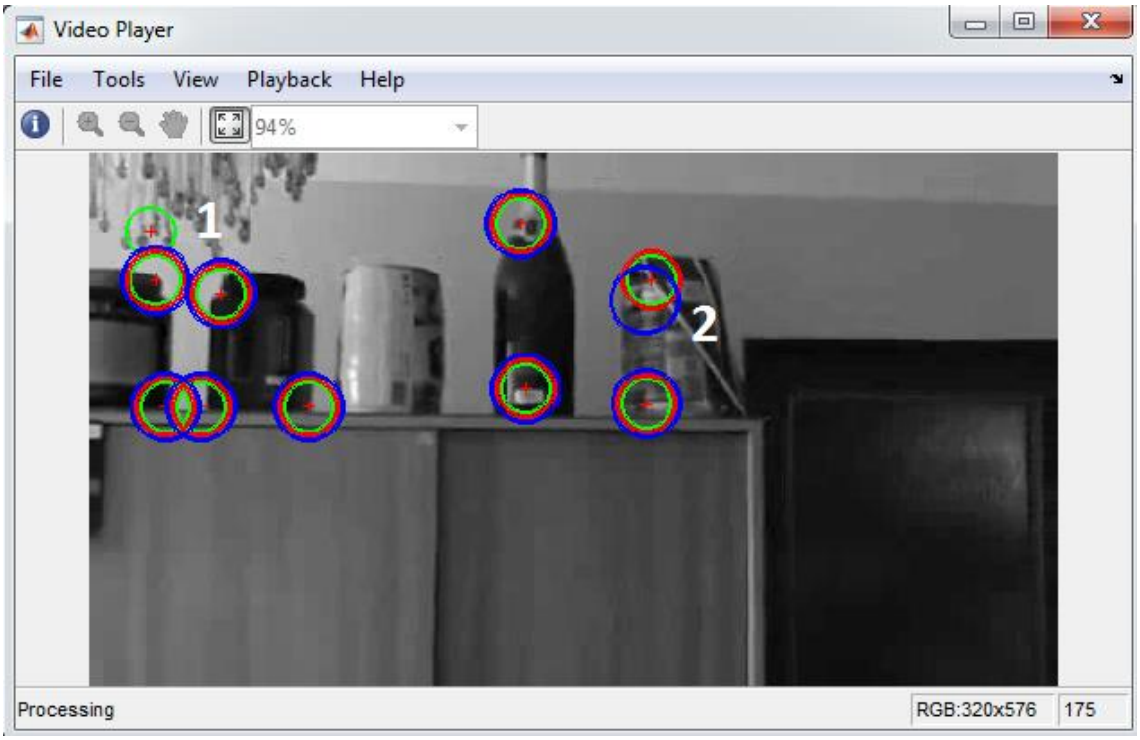
Následně je vykreslen graf, který znázorňuje sumu všech odchylek pro jednotlivé snímky (pro každý snímek součet odchylky). Jak lze vidět na Obrázek č. 14. Graf ukazuje hodnotu odchylek, kdy hodnotám, které byly ztraceny, nebo špatně trasovány je přiřazena maximální možná odchylka (*radius*). Lze tak z grafu vyčíst, ve které obrazu došlo ke ztrátě bodů, případně zvýšení odchylky.



Obrázek č. 14 Suma odchylek pro jednotlivé snímky

Nyní je počítána suma všech odchylek přes všechny snímky tak, abychom v závěru získali pouze jednu hodnotu *deviation_all*, která reprezentuje, jak dobře algoritmus v průběhu celého videa zvládal udržet referenční na jejich definovaných pozicích tak, aby měl co nejmenší odchylku mezi těmito pozicemi.

Poté je znovu spuštěn videopřehrávač, který přehraje znovu celé video, na němž však kromě bodů získaných algoritmem jsou zobrazeny také referenční body, které jsou touto metodou úspěšně trasovány (pouze úspěšné, ne všechny referenční). Pro názornost je na Obrázek č. 15 zobrazen průběh z tohoto videa. Jak lze pozorovat, *zeleně* jsou zobrazeny všechny body, které metoda trasovala (tyto body ovšem nemusí být shodné s referenčními, jak lze vidět na obrázku u *bílého* čísla 1). *Červeně* jsou zobrazeny body, které metoda trasovala, a zároveň jsou shodné s referenčními body. Nakonec *modře* jsou zobrazeny všechny referenční body, které metoda úspěšně trasovala (ale na jejich referenční pozici viz obrázek s *bílým* číslem 2), tím můžeme vidět odchylku mezi referenčním a trasovaným bodem. V ideálním případě se všechny 3 kruhy překrývají, což značí, že je bod správně trasovaný (viz. Obrázek).



Obrázek č. 15 Ukázka pro srovnání trasovaných a referenčních bodů

Poté již následuje úsek výpočtů dalších jednotlivých kritérií hodnocení algoritmů jako například doba úspěšně a neúspěšně trasovaných bodů (v počtu snímků), nebo počet úspěšně trasovaných trajektorií a také procentuální doba, kdy ve videosekvenci byly ztraceny, nebo trasovány (úspěšně i neúspěšně) body.

5.1 Podmínky pro splnění jednotlivých kritérií

Odolnost vůči změně osvětlení:

Algoritmus je odolný vůči změně osvětlení, pokud při náhlé, nebo postupné změně osvětlení nebudou ztraceny všechny trasované body, ale zůstane i nadále skrze tuto změnu trasováno alespoň 25% bodů.

Odolnost vůči motion blur:

Algoritmus je považován za odolný vůči pohybovému zkreslení, pokud bude splněna následující podmínka. Pokud při vzniku pohybového zkreslení nedojde k tomu, že se ztratí trasované body a trasování začne po skončení tohoto jevu. Pokud bude ztraceno větší množství, než 25% bodů je algoritmus považován za neúčinný vůči tomuto vlivu. Další možnosti je částečná odolnost, kdy sice dojde ke ztrátě bodů, ale algoritmus je schopen tyto ztracené body zpětně dohledat, nebo je algoritmus schopen za určitých podmínek toto zkreslení zvládat.

Odolnost vůči změně vzhledu objektu:

Algoritmus je považován za odolný vůči změně vzhledu objektu, pokud je schopen trasovat část objektu navzdory průběžné změně vzhledu tohoto objektu.

Odolnost vůči stínům, odleskům:

Splnění této podmínky je podmíněno tím, že při detekci významného bodu na některém objektu musí být tento bod trasován i v případě, že na tomto objektu bude docházet k vytváření odlesků, případně stínů.

Odolnost vůči geometrickým transformacím:

Odolnost algoritmu je splněna v případě, že při jakýchkoliv definovaných transformacích je algoritmus schopen udržet trasování daného bodu, pokud je to možné (např. dokud nedojde k překročení bodu za okraj obrazu, překrytí okolí).

Odolnost vůči driftu bodů:

Odolnost algoritmu je podmíněna tím, aby při trasování bodů nedocházelo k neustálému putování bodů kolem definované polohy ve snaze nalézt optimální umístění bodu, ale byla udržována stálá poloha na nalezeném místě.

4, Schopnost znovu nalézt ztracený bod: ANO

Pokud dojde ke ztrátě bodu (např. kvůli překrytí) a algoritmus dokáže po objevení se tohoto bodu zpět ve scéně tento bod znovu detekovat a trasovat.

5.2 Kritéria metody KLT

V této podkapitole budou popsány veškerá kritéria hodnocení metody KLT. Pokud bude potřeba, budou k danému kritériu hodnocení popsány podmínky, při kterých je nebo není kritérium správné, nebo odpovídající.

1, Velikost odchylek pro jednotlivé videosekvence (suma odchylek)

Tabulka 4 Odchylky bodů metody KLT

Číslo videosekvence	**Celková odchylka [px]**
Videosekvence 1 (mobil)	$2{,}8040 \cdot 10^3$
Videosekvence 2 (fotoaparát)	$2{,}2813 \cdot 10^3$
Videosekvence 3 (mobil)	$6{,}2252 \cdot 10^3$
Videosekvence 4 (fotoaparát)	$2{,}2884 \cdot 10^4$

2, Počet a doba úspěšně a neúspěšně trasovaných bodů

Tabulka 5 Úspěšnost trasování metody KLT

Číslo videosekvence	Video1	Video2	Video3	Video4
Maximální počet bodů	4820	4800	4590	5100
Počet ztracených bodů	300	199	346	522
Počet trasovaných bodů	4520	4601	4244	4578
Počet úspěšně trasovaných	4435	4558	3926	3784
Počet neúspěšně trasovaných	85	43	318	794
Procentuální doba ztracených	6,22%	4,14%	7,54%	10,24%
Procentuální doba trasování	93,77%	95,86%	92,46%	89,76%
Procentuální doba úspěšného trasování (z doby průběhu trasování)	98,11%	99,06%	92,50%	82,65%

3, Odolnost algoritmů vůči změnám podmínek

Odolnost vůči změně osvětlení: NE

Videosekvence číslo 3 a 4 byla zaměřena na testování schopnosti algoritmů poradit si s náhlou i postupnou změnou osvětlení. Při náhlé změně osvětlení však došlo ke ztrátě všech trasovaných bodů. Teprve při postupném rozsvěcování světel v místnosti docházelo k nalezení významných bodů, které si již algoritmus dokázal udržet, nicméně při ztrátě těchto bodů byly nalezeny i přes silný práh detektoru

špatné body, které poté byly špatně trasovány, dokud se světelné podmínky nezačaly postupně ustalovat.

Odolnost vůči motion blur: ANO

Videosekvence 2 a 4 vykazovaly silnou přítomnost pohybového zkreslení při prudkém otáčení, nebo rychlou chůzí s fotoaparátem. Nicméně tento trasovací algoritmus si ve většině případů dokázal brilantně poradit s tímto problémem, dokonce i při velmi silném zkreslení. Existovaly ale i úseky, kdy kvůli velkému zkreslení došlo ke špatnému určení šablony, a trasovaný bod (a tedy i šablona) se posunul například na hranu trasovaného objektu (nástěnka ve videosekvenci 4), nebo zcela mimo trasovaný prostor (zeď 4. videosekvence). K tomuto ovšem došlo jen v pár případech a při velmi zkreslených snímcích.

Odolnost vůči změně vzhledu objektu: ANO

Za změnu vzhledu objektu lze do jisté míry považovat postupné obejití železné tyče uprostřed místnosti ve videosekvenci číslo 3. Při obcházení lze pozorovat změny vzhledu, jelikož lze nejprve pozorovat, že tyč je na vrcholu zdvojená, ale při postupném obcházení a je jedna část zakrytá. Algoritmus však bod nalezený na vrcholu této tyče úspěšně sleduje po celou dobu, kdy je tyč v záběru (s jednou výjimkou překrytí tyče druhou tyčí).

Odolnost vůči stínům, odleskům: ANO

Ve videosekvencích 1, 2 lze pozorovat výskyt odlesků světla několika nádob. Navzdory tomu, že se odlesk od těchto nádob průběžně mění (zvyšuje se, ztrácí se) je algoritmus i přesto schopen sledovat detekovaný významný bod.

Odolnost vůči geometrickým transformacím: ANO

Především videosekvence 1 a 2 byly zaměřeny na testování geometrických transformací (měřítko, rotace kamery do stran, rotace s kamerou, posuv, zkosení) algoritmus si se všemi pohyby poradil a byl schopen úspěšně trasovat body.

Odolnost vůči driftu bodů: ANO

Algoritmus byl schopen udržet body na jejich výchozí pozici bez toho, aby se během trasování měnila pozice od správného místa bodů. K výrazným posunům bodů docházelo pouze v případě, kdy byla detekována hrana, případně kdy se kvůli zkreslení bod dostal zcela mimo šablonu (videosekvence 3 a 4). Jinak byl schopen udržovat trasovaný bod bez skokových odchylek od správné pozice.

4, Schopnost znovu nalézt ztracený bod: ANO

Ukázka schopnosti znovu nalézt ztracený bod je ve videosekvencích 3 a 4, kde dojde k překrytí kovové tyče jinou tyčí a na několik snímku je tyč ztracena. Nicméně jakmile dojde k jejímu objevení, metoda tento bod nalezne a sleduje.

5, Výpočetní náročnost algoritmů

Tabulka 6 Výpočetní náročnost metody KLT

Číslo videosekvence	Průměrný počet snímků zpracovaných za vteřinu [snímků/s]
Sekvence 1 (mobil, 482 snímků)	48,39
Sekvence 2 (fotoaparát, 480 snímků)	11,17
Sekvence 3 (mobil, 459 snímků)	44,47
Sekvence 4 (fotoaparát, 510 snímků)	10,80

Jak lze vidět z tabulky pro zjištění doby trvání zpracování jednotlivých videí, lze pozorovat, že pro sekvence s nízkým rozlišením (sekvence 1 a 3) dokáží být zpracovány rychleji, než je jejich rychlost snímkování, při kterém byly pořízeny (30 snímků/s). Videa s vyšším rozlišením (2,4) algoritmus zpracovává déle, nicméně i tato rychlost zpracování je v porovnání s ostatními velmi rychlá.

5.3 Kritéria metody IPAN

Podkapitola shrnuje veškerá kritéria hodnocení metody IPAN. Pokud bude potřeba, budou k danému kritériu hodnocení popsány podmínky, při kterých je nebo není kritérium správné, nebo odpovídající.

1, Velikost odchylek pro jednotlivé videosekvence (suma odchylek)

Tabulka 7 Odchylky bodů metody IPAN

Číslo videosekvence	Celková odchylka [px]
Videosekvence 1 (mobil)	$7{,}6698 \cdot 10^4$
Videosekvence 2 (fotoaparát)	$1{,}5729 \cdot 10^5$
Videosekvence 3 (mobil)	$8{,}2692 \cdot 10^4$
Videosekvence 4 (fotoaparát)	$1{,}5258 \cdot 10^5$

Vysoké hodnoty odchylek od referenčních bodů jsou způsobeny především velkým driftem jednotlivých bodů, jelikož metoda nezohledňuje žádnou podobnost s obrazem, ale vychází pouze z detekovaných bodů a podle nich také

hledá nejoptimálnější pozici v aktuálním snímku. Tyto hodnoty odchylek jsou tedy vysoké i navzdory tomu, že byl relativně velký počet bodů ztracen.

2, Počet a doba úspěšně a neúspěšně trasovaných bodů

Tabulka 8 Úspěšnost trasování metodou IPAN

Číslo videosekvence	Video1	Video2	Video3	Video4
Maximální počet bodů	4820	4800	4590	5100
Počet ztracených bodů	1447	2620	1217	2726
Počet trasovaných bodů	3373	2180	3373	2374
Počet úspěšně trasovaných	2934	1923	2570	1388
Počet neúspěšně trasovaných	439	257	803	1026
Procentuální doba ztracených	30,02%	54,58%	26,51%	53,45%
Procentuální doba trasování	69,98%	45,42%	73,49%	46,55%
Procentuální doba úspěšného trasování (z doby průběhu trasování)	86,98%	88,21%	76,20%	58,46%

Velký počet ztracených bodů je způsoben především tím, že metoda je náchylná na pohybové zkreslení a je velmi závislá na schopnosti detektoru významných bodů nalézt nějaké body. A to se i přes velmi nízký práh detektoru při velkém zkreslení nedaří, kdyby byl práh snížen ještě víc, mohly by být detekovány také hrany a místa, kde se žádné opravdu významné body nenachází.

3, Odolnost algoritmů vůči změnám podmínek

Odolnost vůči změně osvětlení: ANO

Ve videosekvenci 3 a 4, které byly zaměřeny na testování těchto vlivů lze pozorovat, že i při náhlé i postupné změně byl algoritmus schopen si udržovat více než 25% bodů, které byly trasovány před touto změnou, i když nebyly součástí referenčních bodů, nicméně jejich pozice byla stálá i tuto přes změnu.

Odolnost vůči motion blur: Částečně

Schopnost algoritmu vypořádat se s pohybovým zkreslením je pouze částečná. Metoda je totiž velmi silně závislá a nalezení významných bodů a tedy i na prahu detektoru, nicméně při přítomnosti pohybového zkreslení je velmi obtížné nalézt nějaké body, které by se daly trasovat. Jelikož při tomto zkreslení i pokud jsou body nalezeny, nedají se příliš úspěšně a dlouze trasovat. V momentě kdy k tomuto zkreslení dochází, jsou body trasované metodou většinou ztraceny a to i přes velmi nízký práh. Metoda však řeší problém pohybového zkreslení v *post-processingu,*

kdy jsou zpětně hledány body, které se nachází uvnitř spojené trajektorie, která obsahuje „díry“ v podobě těchto ztracených bodů. Jedná se o zpětné spojování trajektorie, která ovšem vyžaduje neustálé hledání nových bodů v každém snímku (v celém snímku maximální počet bodů), pokud dojde k jejich ztrátě. Z těchto bodů si poté algoritmus zvolí ty, které odpovídají nastaveným parametrům. Výsledky této metody jsou tedy zkreslené, protože při ztrátě bodů algoritmus nehledá aktivně nové body, ale čeká až na příchod nově nalezených referenčních bodů, které pokud je trajektorie dlouhá mohou přijít až za dobu, kdy nepůjde trajektorii zpětně doplnit (omezeno na 6 snímků). Proto metoda s aktivním vyhledáváním bodů vykazuje lepší výsledky. Nicméně kvůli zajištění stejných podmínek pro každý algoritmus je hledání nových bodů omezeno. Z tohoto důvodu algoritmus nemůže doplnit chybějící trajektorie, jelikož ve většině případů ani nedojde k vytvoření těchto „děr“ v trajektorii.

Odolnost vůči změně vzhledu objektu: ANO částečně

Algoritmus je schopný trasovat významný bod některého objektu i přes postupnou změnu vzhledu tohoto objektu, pokud při této změně vzhledu však nedojde ke ztrátě bodu (pohybovým zkreslením). Pokud je splněna podmínka, že k této ztrátě bodu nedojde, je algoritmus schopen tento bod sledovat.

Odolnost vůči stínům, odleskům: ANO

Ve videosekvenci 1 a 2, kde k tomuto jevu dochází lze pozorovat, že i přes značný drift bodů v okolí kolem objektu, kde nastal odlesk je algoritmus schopen úspěšně sledovat tento bod. Toho trasování probíhá úspěšně i přes postupné rozšiřování a zkracování se odlesku od předmětu.

Odolnost vůči geometrickým transformacím: ANO

Algoritmus byl schopen i přes působící geometrické transformace způsobené různými pohyby schopen udržet pozici významných bodů.

Odolnost vůči driftu bodů: NE

Ve všech pořízených videosekvencích lze pozorovat velký výskyt driftu bodů. Je to způsobeno tím, že metoda vypočítává pozici nového bodu k trasování pouze pomocí penalizace bodu při změně směru a velikosti. Pokud je tedy nalezen jen malý počet bodů v možném okolí, nebo pokud je nalezen pouze jeden bod v přípustném okolí, je bod považován za součást trajektorie trasovaného.

4, Schopnost znovu nalézt ztracený bod: ANO

Ve videosekvencích je možno pozorovat, že při překrytí objektu, na kterém byl trasován bod, dojde k jeho odkrytí k jeho následnému sledování. Pokud však nedojde k jeho trasování ihned, ale až po určité chvíli, je možné, že bod bude trasován zpětně po dokončení post-processingu ztracených trajektorií.

5, Výpočetní náročnost algoritmů

Tabulka 9 Výpočetní náročnost metody IPAN

Číslo videosekvence	Průměrný počet snímků zpracovaných za vteřinu [snímků/s]
Sekvence 1 (mobil, 482 snímků)	15,71
Sekvence 2 (fotoaparát, 480 snímků)	3,69
Sekvence 3 (mobil, 459 snímků)	14,65
Sekvence 4 (fotoaparát, 510 snímků)	3,90

Jak lze pozorovat z výše uvedené tabulky je výpočetní náročnost zpracování jednotlivých snímků podstatně větší, než u metody KLT. Pro sekvence s nízkým rozlišením je algoritmus schopen zpracovávat snímky s poloviční rychlostí, než se kterou byly pořízeny. Pro sekvence s vyšším rozlišením je rychlost zpracování podstatně nižší a to až 10x pomalejší, než rychlost, kterou byly snímky pořízeny.

5.4 Kritéria metody RC (rank constrained)

Podkapitola shrnuje veškerá kritéria hodnocení metody RC. Pokud bude potřeba, budou k danému kritériu hodnocení popsány podmínky, při kterých je nebo není kritérium správné, nebo odpovídající.

1, Velikost odchylek pro jednotlivé videosekvence (suma odchylek)

Tabulka 10 Odchylky bodů metody RC

Číslo videosekvence	Celková odchylka [px]
Videosekvence 1 (mobil)	$4{,}5622 \cdot 10^4$
Videosekvence 2 (fotoaparát)	$8{,}6001 \cdot 10^4$
Videosekvence 3 (mobil)	$3{,}6232 \cdot 10^4$
Videosekvence 4 (fotoaparát)	$8{,}9563 \cdot 10^4$

2, Počet a doba úspěšně a neúspěšně trasovaných bodů

Tabulka 11 Úspěšnost trasování metodou RC

Číslo videosekvence	Video1	Video2	Video3	Video4
Maximální počet bodů	4820	4800	4590	5100
Počet ztracených bodů	955	2639	990	2608
Počet trasovaných bodů	3865	2161	3600	2492
Počet úspěšně trasovaných	2871	1762	2873	1825
Počet neúspěšně trasovaných	994	399	727	667
Procentuální doba ztracených	19,81%	54,98%	21,57%	51,14%
Procentuální doba trasování	80,19%	45,02%	78,43%	48,86%
Procentuální doba úspěšného trasování (z doby průběhu trasování)	74,28%	81,53%	79,80%	73,23%

Velký počet ztracených bodů je způsoben pomalý a postupných driftem bodů z jejich výchozích pozic v důsledku skupinového pohybu všech bodů. Jelikož také nebyla určena strategie vyřazování bodů, byl zvolen pro každou šablonu velmi nízkých práh detektoru významných bodů, kdy musí být nalezen alespoň jeden bod v celé šabloně, aby nedošlo k jejímu vyřazení. Z tohoto důvodu dochází při větší odchylce od pozice významného bodu k vyřazení tohoto bodu z trasování. Poté musí dojít k nalezení dalšího referenčního bodu v aktuálním, nebo následujících snímcích k započetí dalšího trasování.

3, Odolnost algoritmů vůči změnám podmínek

Odolnost vůči změně osvětlení: ANO, částečně

Algoritmus ve videosekvenci č. 3 sice dokázal při prudké změně osvětlení udržet více než 25% trasovaných bodů, ovšem ve videosekvenci č. 4 došlo ke ztrátě všech trasovaných bodů. Z tohoto důvodu algoritmus považovat za částečně odolný vůči změně osvětlení. Odolnějším by se mohl algoritmus stát v případě, že i v průběhu prudké změny osvětlení by byl schopen nalézt alespoň pár silných bodů, které by odolaly vyřazovací podmínce. V tomto případě by došlo k tomu, že by silné body dokázaly udržet pozice slabších bodů (pokud by nedošlo k jejich vyřazení). Tyto podmínky však nelze vždy zaručit.

Odolnost vůči motion blur: Částečně ANO

Schopnost algoritmu odolávat vlivům pohybového zkreslení je podmíněna existenci alespoň několika silných bodů, které by dokázaly udržet pozice slabších bodů na správném místě. Pokud tedy nedochází k příliš velkému zkreslení, jaké je pozorovatelné například u videosekvencí č. 4. Je algoritmus při výskytu menšího zkreslení a zajištění alespoň jednoho silného bodu na jeho základě posouvat ostatní body do nejoptimálnějších pozic. Nicméně i tento posuv je většinou nesprávných, jelikož se algoritmus pohybuje pouze podle pár silných bodů, a tím zanedbává některé geometrické transformace (hlavně u vzdálenějších bodů tím dochází ke značné chybě v posunu).

Odolnost vůči změně vzhledu objektu: ANO

Ve videosekvencích č. 3 a 4 kde docházelo k největším změnám vzhledu objektu (kvůli obcházení kolem objektu) lze pozorovat, že algoritmus je schopen sledovat body i přes tuto změnu. Trasování již zmíněné tyče, která průběžně mění vzhled, sice nevyšlo, kvůli předčasné ztrátě tohoto bodu, ale při sledování železné konstrukce za touto tyčí potvrdilo schopnost metody tuto změnu respektovat. Při celkovém obcházení této konstrukce byl algoritmus schopen sledovat body nacházející se na ní nezávisle na vzhledu.

Odolnost vůči geometrickým transformacím: ANO, částečně

V průběhu trasování především ve videosekvencích 1 a 2 lze pozorovat nejčastěji pro situace, kdy je kamerou nakláněno do stran, že spousta bodů přestává reagovat být trasována a vzdaluje se od tvých výchozích pozic. Je to způsobeno tím, že algoritmus je vůči geometrickým transformacím odolným jen částečně, a to z toho důvodu, že posouvání všech bodů je prováděno jen podle nejsilnějších a nejvíce padnoucích bodů do jejich šablon. V tomto případě tedy dochází k tomu, že je pohyb ostatních bodů závislý na několika velmi silných bodech, kde došlo k nejmenší obrazové chybě a podle těchto bodů je pohybováno se všemi. V tom případě tedy dochází k velkým chybám, jelikož zanedbáváme různé vzdálenosti od kamery (měřítko, z-osa) a pohybuje všemi body stejně. Jisté vylepšení by také (hlavně pro změnu měřítka) bylo použití pyramidové implementace, která dokáže tyto pohyby zachytit. Nicméně toto zanedbávání různých vzdáleností je největší nevýhodou tohoto algoritmu, jelikož dochází k velkým chybám. Jisté zlepšení tohoto problému by bylo nastavit parametr *total=0*, kdy algoritmus posouvá každým bodem zvlášť. Nelze ovšem tvrdit, že algoritmus vůči těmto transformacím odolný není, jen posouvá všemi body podle těchto transformací v závislosti na několika bodech.

Odolnost vůči stínům, odleskům: ANO

Jak lze pozorovat na videosekvenci 1 a 2 kde je přítomen vznik odlesků na objektech. Je algoritmus schopen tomuto jevu odolávat a při přes vznik a zánik tohoto odlesku sledovat nalezený významný bod.

Odolnost vůči driftu bodů: NE

Algoritmus je náchylný na driftování bodů. Je to způsobeno především zanedbávání různých vzdáleností jednotlivých bodů a posouváním všech bodů jedním směrem ignorováním těchto vzdáleností. Dochází tedy poté k drobným odchylkám od správné pozice, které se ovšem s každým snímkem většinou zvětšují, poté je kvůli zvýšené odchylce také špatně posunutá šablona což vede opět k chybě, jelikož algoritmus naleze nejvhodnější pozici, ovšem pro nepřesnou šablonu. Řešením toho problému by bylo jemnější a přesnější posouvání šablon (na sub-pixelové úrovni). Nicméně i při tomto řešení by docházelo k pomalému driftování bodů z důvodu zanedbávání vzdálenosti.

4, Schopnost znovu nalézt ztracený bod: ANO

Na videosekvencích 3 a 4, kde dochází k chvilkové ztrátě bodů z důvodu překrytí, lze pozorovat schopnost algoritmu vypořádat se s tímto problémem. Nejlépe lze vidět tuto schopnost opět na již zmíněné konstrukci, kterou průběžně překrývají 2 tyče (na různou dobu). V této situaci lze vidět velkou sílu tohoto algoritmu, jelikož je schopen v závislosti na ostatních bodech posouvat pozici i chvilkově ztraceného bodu tak, aby při jeho objevení byla nalezena jeho pozice. Toto ovšem platí pro případ, kdy detektor nevyřadí chvilkově ztracený bod. Pokud ovšem k vyřazení nedojde lze trasovat bod i při dočasné ztrátě ze snímku.

5, Výpočetní náročnost algoritmů

Číslo videosekvence	**Průměrný počet snímků zpracovaných za vteřinu [snímků/s]**
Sekvence 1 (mobil, 482 snímků)	2,97
Sekvence 2 (fotoaparát, 480 snímků)	3,13
Sekvence 3 (mobil, 459 snímků)	3,09
Sekvence 4 (fotoaparát, 510 snímků)	3,26

Jak lze vidět z výše uvedené tabulky, je tato implementovaná metoda z ostatních nejpomalejší. Rychlost je závislá na volbě velikosti šablon, počtu posunutí šablon do různých směrů, délce historie a dalších parametrů. Výrazného zrychlení by bylo dosaženo pomocí iterační metody, která by snížila počet posouvání šablony.

5.5 Srovnání implementovaných metod

V této sekci budou srovnány jednotlivé metody z hlediska jejich vlastností a výhod a nevýhod, případně bude uveden příklad, ve kterém by bylo vhodné použít jednotlivou metodu. Bude zde také uvedena tabulka, která shrnuje jednotlivé parametry použitých metod do jedné pro všechny videosekvence. A také tabulka, shrnující jednotlivé odolnosti vůči nepříznivým testovaným vlivům.

Tabulka 12 Tabulka srovnání parametrů stability všech metod

Parametr	**Metoda**	**Číslo videosekvence**			
		1	**2**	**3**	**4**
Celková odchylka [px]	**KLT**	$2{,}8 \cdot 10^3$	$2{,}3 \cdot 10^3$	$6{,}2 \cdot 10^3$	$2{,}3 \cdot 10^4$
	IPAN	$7{,}7 \cdot 10^4$	$1{,}6 \cdot 10^5$	$8{,}3 \cdot 10^4$	$1{,}5 \cdot 10^5$
	RC	$4{,}6 \cdot 10^4$	$8{,}6 \cdot 10^4$	$3{,}6 \cdot 10^4$	$8{,}9 \cdot 10^4$
Procentuální doba ztracených	**KLT**	6,22%	4,14%	7,54%	10,24%
	IPAN	30,02%	54,58%	26,51%	53,45%
	RC	19,81%	54,98%	21,57%	51,14%
Procentuální doba trasování	**KLT**	93,77%	95,86%	92,46%	89,76%
	IPAN	69,98%	45,42%	73,49%	46,55%
	RC	80,19%	45,02%	78,43%	48,86%
Procentuální doba úspěšného trasování	**KLT**	98,11%	99,06%	92,50%	82,65%
	IPAN	86,98%	88,21%	76,20%	58,46%
	RC	74,28%	81,53%	79,80%	73,23%
Průměrný počet snímků zpracovaných za vteřinu [snímků/s]	**KLT**	48,39	11,17	44,47	10,80
	IPAN	15,71	3,69	14,65	3,90
	RC	2,97	3,13	3,09	3,26

Další parametry pro kompletní srovnání jsou uvedeny níže.

Tabulka 13 Celkové srovnání parametrů robustnosti algoritmů

Parametr	Metoda, hodnocení		
	KLT	**IPAN**	**RC**
Odolnost vůči změně osvětlení	**NE**	**ANO**	**Částečně**
Odolnost vůči motion blur	**ANO**	**Částečně**	**Částečně**
Odolnost vůči změně vzhledu objektu	**ANO**	**Částečně**	**ANO**
Odolnost vůči stínům, odleskům	**ANO**	**ANO**	**ANO**
Odolnost vůči geometrickým transformacím	**ANO**	**ANO**	**Částečně**
Odolnost vůči driftu bodů	**ANO**	**NE**	**NE**
Schopnost znovu nalézt ztracený bod	**ANO**	**ANO**	**ANO**

5.5.1 Hodnocení metody KLT

Jak lze pozorovat z výše uvedených tabulek, nejlepších výsledků ve všech parametrech, ohledně stability a výpočetní náročnosti dosahuje metoda KLT. Je to částečně způsobeno i tím, že referenční body byly tvořeny pomocí této metody a nevhodné body byly ořezány. Navzdory tomuto faktu však metoda stále vykazuje ve všech směrech nejlepších výsledků. A to i co se týče délky udržení trasovaných bodů. Metoda je schopna udržet trasovaný bod po dlouhou dobu trvání videosekvence, většinou až do doby, než dojde ke ztrátě bodu za okraj obrazu, nebo pokud je ztracena jeho poloha nejčastěji kvůli silnému pohybovému zkreslení. Metoda dosahovala také nejmenších procent ztracených bodů a zároveň nejpřesnější úspěšné sledování jednotlivých bodů s nejmenšími odchylkami. Kromě náhlých změn osvětlení v místnosti metoda vykazuje i silné kladné vlastnosti robustnosti, se kterými si dokázala poradit. Metoda také vykazuje nejlepších výsledků ve výpočetní náročnosti, jelikož je optimalizovaná pomocí pyramidální implementace a iteračních metod k zajištění co nejrychlejší konvergence pro nalezení optimální pozice v aktuálním snímku. Algoritmus je vhodný použít například pro video stabilizaci, případně odhad pohybu kamery a trasování objektu. Tato metoda je i přes dlouhou dobu, od svého publikování, díky svým výsledkům stále jednou z nejpoužívanějších metod pro trasování.

5.5.2 Hodnocení metody IPAN

Největší nevýhodou této metody je, že je silně závislá na detektoru významných bodů, a také na nastavení svých parametrů. Metoda používá k nalezení nové vhodné pozice hodnotící funkce, která penalizuje změny ve směru a velikosti trajektorie bodu složené vždy z 3 bodů, které postupně tvoří trajektorii. Metoda tedy nezohledňuje žádnou podobnost bodu v předchozích snímcích a pouze na základě nalezených bodů volí nejoptimálnější polohu aktuálního bodu. Pokud tedy dojde k nalezení pouze jednoho bodu v dané vymezené oblasti, bude bod považován za optimální, jelikož nebude mít konkurenta. Toto chování je také důsledkem nejvyšších hodnot odchylek od referenčních bodů, jelikož dochází k driftování bodů podle počtu nalezených bodů. Jelikož je metoda náchylná na pohybové zkreslení, dochází při tomto jevu většinou ke ztrátě bodů, jelikož detektor bodů i přes nízký práh většinou není schopen nalézt bod, který by se dal trasovat. Algoritmus je sice schopen se vypořádat s tímto jevem v post-processingu, nicméně délka trvání tohoto jevu je omezena na 5 snímků, poté již algoritmus nedokáže spojit přerušené trajektorie, jelikož se za tuto dobu může velmi změnit pozice významného bodu i s předpokladem maximální rychlosti. . Nicméně pokud je k dispozici v každém snímku dostatek vhodných bodů je schopen algoritmus úspěšně sledovat body bez větších odchylek. K tomuto však potřebuje kvalitní videosekvence, na nichž má pohybové zkreslení vliv jen minimálně, nejlépe vůbec. Výpočetní doba je výrazně pomalejší, než u KLT, nicméně je rychlejší než u metody RC. Pro videa s malým rozlišením je výpočetní rychlost vyšší. Vyšší doba výpočtu je způsobená tím, že metoda v každém snímku hledá neomezené množství bodů s velmi nízkým prahem a algoritmus pro nalezení optimálních bodů probíhá v několika cyklech a to všechny body v definované oblasti. Nicméně si algoritmus dokáže poradit se změnami světelných podmínek, díky nízkému prahu detektoru a tím nalezení vhodných bodů v okolí před touto změnou. Stejně tak lze sledovat body, na kterých se mohou vyskytovat odrazy, odlesky a je také odolný vůči geometrickým transformacím, jelikož nové body je hledají detektorem a nedochází ke kolizím (např. při změně měřítka) mezi dvěma šablonami. Výhodou je také vlastnost algoritmu nalézt již jedno ztracený bod, díky své sekci post-processingu, která umožňuje spojit díry uvnitř jednotlivých trajektorií zpětně. Toto je jednou z největších výhod, kterou autoři uvádí a navrhují použití své metody nejčastěji pro videosekvence, kde k takovým dočasným ztrátám objektu dochází např. pro bezpečnostní monitorování, nebo správu video databází.

5.5.3 Hodnocení metody RC

Metoda je založená na myšlence společného stavu a posunu bodů na základě minimalizace obrazové funkce a penalizační funkce a nalezení minima této energie. Nejdůležitějším předpokladem pro správnou funkci tohoto algoritmu je existence alespoň pár silných bodů, které budou diktovat směr posunu i pro slabší body, jejichž posun může obecně vyhovovat více směrům (hrany), v tomto případě metoda vykazuje velmi silné výsledky, protože například ve srovnání s metodou KLT by při detekci a trasování hrany objektu bod značně driftoval. Při zajištění těchto podmínek však algoritmus dokáže sledovat i hrany objektu mez driftu bodů. Nicméně vzhledem k tomu, že je k jednotlivým bodům přistupováno jako k jedné skupině, která se pohybuje v jednom směru dochází k ignorování vzdáleností mezi jednotlivými body. Což vede ke správnému posunu většinou pouze nejsilnějších bodů a posun pro ostatní body, které se nachází na jiných vzdálenostech od kamery, dochází ke špatnému posunu, což má za následek drift bodů po malých vzdálenostech od správné pozice, který se však postupně zvětšuje. Ignorování této vzdáleností je největší nevýhodou této metody. Zamezení ignorování těchto geometrických transformací dojde při nastavení proměnné *total=0*, která bude pohybovat každým bodem zvlášť a zohlední tak různé vzdálenosti od kamery. V tom případě však budeme přistupovat k bodům jednotlivě, místo jako k jednomu celku. Při použití této metody dochází k menší ztrátě bodů, než metodou IPAN a i přes fakt, že je úspěšnost trasování vyšší u metody IPAN, je procentuální úspěšnost ovlivněna počtem trasovaných bodů. Z toho vyplývá, že i když má metoda RC nižší úroveň úspěšného trasování, díky vyššímu počtu trasovaných bodů lze říci, že úspěšnost je podobná, případně vyšší. Podobně je to i s velikostí odchylky od referenčních bodů, kdy metoda také vykazuje lepších výsledků. Algoritmus je díky těmto vlastnostem také schopen znovu nalézt ztracený bod a odolávat odleskům a dokonce i změnám světelných podmínek, ovšem musí být vždy přítomen pár silných bodů a přizpůsoben práh pro odstranění špatných bodů. V těchto případech lze odolávat i pohybovému zkreslení, ovšem při přítomnosti těchto bodů, v opačném případě dochází k nekontrolovatelnému pohybu bodů jedním směrem zcela mimo správné pozice. Nicméně výpočetní náročnost této metody je velká, i když při použití pyramidální implementaci a iteračních metod by se dalo docílit řádově lepších výsledků. Algoritmus tedy lze použít i pro videosekvence, které vykazují nižší kvalitu a slabé body, ovšem za podmínek, že budou existovat silné body, které tyto body dokáží udržet.

6 ZÁVĚR

V základní části této práce byly popsány obecné a zásadní pojmy, které souvisejí s problematikou trasování bodů (videosekvence, vektor pohybu, předpoklady pohybu tuhých těles). Dále byly popsány situace, které mohou nastat při pohybu objektu, případně kamery. Byl proveden popis scény, a jevů, které mohou vzniknout při pohybu kamery držené rukou (motion blur).

Dále v této práci byly popsány metody pro trasování objektu a především také metody pro trasování významných bodů, což bylo hlavní náplní této části. Jednotlivé trasovací metody byly stručně popsány, přičemž metoda pro trasování významných bodů byla popsána podrobněji. V práci poté byla vypracována tabulka s dostupnými články, které se týkají metod pro trasování a které byly zpracovány v rámci literární rešerše. Na základě této rešerše poté byly zvoleny 3 principiálně odlišné metody pro trasování významných bodů (KLT, IPAN, RC) s rozestupem kolem 10 let mezi publikacemi.

Další částí bylo vytvoření kritérií, kterými budou implementované metody hodnoceny. Kritéria byla především zaměřena na testování stability a robustnosti jednotlivých algoritmů, stejně jako výpočetní náročnosti. Byly také vytvořeny 4 videosekvence, které měly rozdílné parametry a byly určeny pro testování zvolených kritérií. Byl také ukázán vliv pohybového zkreslení na kvalitu po sobě jdoucích snímcích ve videosekvenci. Následně přišla na řadu samotná implementace všech tří zvolených algoritmů. Po úspěšné implementaci byl vytvořen popis veškerých metod spolu s nastavitelnými parametry a důsledky, které vyplývají ze změny jednotlivých parametrů. Následně byly zvoleny nejlepší parametry pro jednotlivé metody, se kterými byly testovány.

Další částí bylo vytvoření referenčních bodů, které byly vytvořeny pomocí metody KLT a ručně odstraněny body, které se nenacházely na správných pozicích (např. vlivem změny osvětlení atd.). Po vytvoření těchto referenčních bodů byly testovány algoritmy postupně podle všech zvolených kritérií a byly také popsány podmínky, při kterých algoritmus pracuje správně, bylo-li to potřeba.

Následně byla vytvořena **Tabulka 12** a **Tabulka 13**, která shrnovala dosažené výsledky všech algoritmů a umožnila tak jejich porovnání z tohoto hlediska. Po srovnání těchto metod byly zhodnoceny klady a zápory jednotlivých metod s definováním podmínek pro správnou funkčnost, případně možnosti použití těchto metod v různých oblastech, nebo způsoby možného vylepšení. Implementované algoritmy se nachází v přílohách spolu s dodatečnou složkou, ve které byly tvořeny referenční body.

Literatura

[1] Prokšová, J.: *Optické klamy a teorie barevného vidění* [online]. Plzeň, 1970 [cit.2016-1-1], aktualizováno 2008-10-1. Dostupné na URL: < http://www.kof.zcu.cz/st/sm/fpv/doplnek_syl1.doc>

[2] Xiao, J.: SFMedu: *A structure from motion system for education* [online].Princeton [cit.2016-1-1], aktualizováno 2015-11-25. Dostupné na URL: < http://vision.princeton.edu/courses/SFMedu/>

[3] Lindeberg, T.: *Scale selection properties of generalized scale-space interest point detectors*, Journal of Mathematical Imaging and Vision, Svazek 46, Vydání 2, Strany 177-210, 2013.

[4] Horák, K..: Zpracování vícerozměrných signálů [online]. Brno, 2010 [cit.2016-1-1], aktualizováno 2015-11-9. Dostupné na URL: <http://midas.uamt.feec.vutbr.cz/ZVS/Lectures/07_Detekce_hran_a_rohu.pdf>

[5] Wikipedia, The Free Encyclopedia *Feature detection (computer vision)* [online]. 2015 [cit.2016-1-1], aktualizováno 2015-4-3. Dostupné na URL: <https://en.wikipedia.org/wiki/Feature_detection_%28computer_vision%29 >

[6] Horák, K.: Počítačové vidění [online]. Brno, 2010 [cit.2016-1-1], aktualizováno 2015-12-2. Dostupné na URL: <http://midas.uamt.feec.vutbr.cz/POV/Lectures/10_Analyza_pohybu.pdf>

[7] Horák, K.: Aplikace počítačového vidění [online]. Brno, 2010 [cit.2016-1-1], aktualizováno 2015-2-3. Dostupné na URL: <http://midas.uamt.feec.vutbr.cz/APV/lecturespdf/08_Trasovani_objektu.pdf>

[8] JAN, Jiří. *Medical image processing, reconstruction and restoration: concepts and methods. Boca Raton:* Taylor, 2006, 730 s. ISBN 0-8247-5849-8.

[9] BURKHART, John. *The Videomaker Guide to Video Production.* Fourth edition. Burlington USA : Elsevier Inc., 2008. 377 s. ISBN 978-0-240-80968-7.

[10] WU, Jiagu, Zhenzhen ZHENG, Huajun FENG, Zhihai XU, qi LI a Yueting CHEN. *Restoration of TDI camera images with motion distortion*

and blur. Optics and Laser Technology [online]. 2010, 42(8): 1198-1203 [cit. 2016-01-02]. DOI: 10.1016/j.optlastec.2010.03.010. ISSN 0030-3992.

[11] Meilland, M., Drummond, T. ; Comport : *A Unified Rolling Shutter and Motion Blur Model for 3D Visual Registration* [online]. 2013, strany 2016 - 2023 [cit. 2016-01-02]. DOI: 10.1109/ICCV.2013.252. ISSN 1550-5499.

[12] ŠONKA, Milan, Václav HLAVÁČ a Roger BOYLE. *Image processing, analysis, and machine vision*. 3rd ed. Toronto: Thomson, 2008, xxv, 829 s. : il. ISBN 978-0-495-08252-1.

[13] HRNČÍŘ, Zdeněk. *Optický tok v obrazových datech živých buněk* [online]. Brno, 2006 [cit. 2016-01-02]. Diplomová práce. Masarykova univerzita, Fakulta informatiky. Vedoucí práce: Vladimír Ulman, Dostupné z: <http://www.is.muni.cz/th/60514/fi_m/dp.pdf>.

[14] BRADSKI, Gary R a Adrian KAEHLER. *Learning OpenCV*. Sebastopol: O'Reilly, 2008, xvii, 555 s. ill. ISBN 978-0-596-51613-0.

[15] BOVIK, Al. *Handbook of image and video processing*. 2nd ed. Burlington: Elsevir Academic Press, 2005, xv, 1372 s. ISBN 0-12-119792-1.

[16] YOKOYAMA, M. a T. POGGIO. *A Contour-Based Moving Object Detection and Tracking*. In: Visual Surveillance and Performance Evaluation of Tracking and Surveillance, *2005. 2nd Joint IEEE International Workshop on* [online]. USA: IEEE, 2005, s. 271-276 [cit. 2016-01-02].DOI: 10.1109/VSPETS.2005.1570925. ISBN 0-7803-9424-0.

[17] SAUNIER, N. a T. SAYED. A feature-based tracking algorithm for vehicles in intersections. In: *Computer and Robot Vision, 2006. The 3rd Canadian Conference on* [online]. IEEE, 2006, s. 59-59 [cit. 2016-01-02]. DOI: 10.1109/CRV.2006.3. ISBN 0-7695-2542-3.

[18] JIANBO SHI a TOMASI. Good features to track. In: *Computer Vision and Pattern Recognition, 1994. Proceedings CVPR '94., 1994 IEEE Computer Society Conference on* [online]. IEEE Publishing, 1994, s. 593-600 [cit. 2016-01-02]. DOI: 10.1109/CVPR.1994.323794. ISBN 0-8186-5825-8. ISSN 1063-6919.

[19] Carlo Tomasi and Takeo Kanade. Detection and Tracking of Point Features. *Carnegie Mellon University Technical Report CMU-CS-91-132*, April 1991.

[20] A. Shokurov, A.Khropov, and D. Ivanov, *Feature Tracking in Images and Video*, in International Conference on Compur graphics between Europe and Asia (Graphi-Con 2003) (Moscow, Russia, September 2003), 177-179

[21] LOUTAS, E., I. PITAS a C. NIKOU. *Probabilistic multiple face detection and tracking using entropy measures.* Circuits and Systems for Video Technology, IEEE Transactions on [online]. USA: IEEE, 0040n. l., 14(1): 128-135 [cit. 2016-01-02]. DOI: 10.1109/TCSVT.2003.819178. ISSN 1051-8215.

[22] Vanessa Robles, Enrique Alegre, Jose M. Sebastian: *Tracking Algorithms Evaluation in Feature Points Image Sequences*. International Conference, ICIAR 2004, Porto, Portugal, September 29 - October 1, 2004, Proceedings, Part II pp 589-596. ISSN 0302-9743, DOI: 10.1007/978-3-540-30126-4_72

[23] Dmitry Chetverikov, Feature Point Tracking Algorithms [online]. Image and Pattern Analysis, Group Computer and Automation Research Institute Budapest, Kende u.13-17, H-1111 HUNGARY. 1998 [cit.2016-1-1],URL: <http://homepages.inf.ed.ac.uk/rbf/CVonline/LOCAL_COPIES/CHETVERIKOV/psmweb/psmweb.html>

[24] BAKER, Simon a Iain MATTHEWS. Lucas-Kanade 20 Years On: A Unifying Framework. International Journal of Computer Vision [online]. Boston: Kluwer Academic Publishers, 0040n. l., 56(3), 221-255 [cit. 2016-01-06]. DOI: 10.1023/B:VISI.0000011205.11775.fd. ISSN 0920-5691.

[25] Bruce D. Lucas and Takeo Kanade. An Iterative Image Registration Technique with an Application to Stereo Vision. International Joint Conference on Artificial Intelligence, pages 674–679, 1981.

[26] D. Chetverikov and J. Verestóy. Tracking feature points: A new algorithm. In Proc. International Conf. on Pattern Recognition, pages 1436-1438,1998.

[27] D. Chetverikov and A. Lerch. A matching algorithm for motion analysis of dense populations. *Pattern Recognition Letters*, 11:743-749, 1990.

[28] I. K. Sethi and R. Jain. Finding trajectories of feature points in a monocular image sequence. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 9:56-73, 1987.

[29] POLING, Bryan, Gilad LERMAN a Arthur SZLAM. *Better Feature Tracking Through Subspace Constraints* [online]. 2014 [cit. 2016-05-14].

Seznam zkratek

MB Motion Blur

Seznam příloh

A Obsah přiloženého CD

B ZIP soubor *METODY* obsahující jednotlivé algoritmy odevzdaný do IS

A Obsah přiloženého CD

Složka samostatné vyhledávání
- BERUfotak1seda.mp4
    - BERUfotak2seda.mp4
    - BERUmobil1seda.mp4
    - BERUmobil2seda.mp4
    - HANDreferfotak1.mat
    - HANDreferfotak2.mat
    - HANDrefermobil1.mat
    - HANDrefermobil2.mat
    - Složka IPAN
        - DeviationOfRefer_last.m
        - FillHole.m
        - FindAllCandidateInitial.m
        - FindAllCandidateLoop.m
        - findbestcandidateInitial.m
        - findbestcandidateLoop.m
        - FindBestPoint.m
        - FindROI.m
        - ChooseBestRow.m
        - MAIN_IPAN.m
        - MainOtherTrajectoryDetect.m
        - MergeTraject.m
        - Pointsuminitial.m
        - Pointsumloop.m
        - resizepoint.m
        - startendtrajectory.m
        - TestHypotesis.m
        - Trajectory2.m
    - Složka KLT
        - DeviationOfRefer_last.m
        - findbestpoint3.m
        - MAIN_KLT.m
        - startendtrajectory.m
        - Trajectory2.m
    - Složka RC
        - DeviationOfRefer_last.m
        - findbestpoint3.m
        - MAIN_RC.m
        - startendtrajectory.m
        - Trajectory2.m
        - Evaluate_P.m
        - Evaluate_Energy.m
        - findROIActualTemplate.m
        - findROIoldTemplate.m
        - genMesh.m

Diplomová práce_el.verze.pdf

Složka – s referenčními body
- BERUfotak1seda.mp4
- BERUfotak2seda.mp4
- BERUmobil1seda.mp4
- BERUmobil2seda.mp4

- HANDreferfotak1.mat
- HANDreferfotak2.mat
- HANDrefermobil1.mat
- HANDrefermobil2.mat
- Složka IPAN
    - DeviationOfRefer_last.m
    - FillHole.m
    - FindAllCandidateInitial.m
    - FindAllCandidateLoop.m
    - findbestcandidateInitial.m
    - findbestcandidateLoop.m
    - FindBestPoint.m
    - FindROI.m
    - ChooseBestRow.m
    - MAIN_IPAN.m
    - MainOtherTrajectoryDetect.m
    - MergeTraject.m
    - Pointsuminitial.m
    - Pointsumloop.m
    - resizepoint.m
    - startendtrajectory.m
    - TestHypotesis.m
    - Trajectory2.m
- Složka KLT
    - DeviationOfRefer_last.m
    - findbestpoint3.m
    - MAIN_KLT.m
    - startendtrajectory.m
    - Trajectory2.m
- Složka RC
    - DeviationOfRefer_last.m
    - findbestpoint3.m
    - MAIN_RC.m
    - startendtrajectory.m
    - Trajectory2.m
    - Evaluate_P.m
    - Evaluate_Energy.m
    - findROIActualTemplate.m
    - findROIoldTemplate.m
    - genMesh.m
- Složka tvorba refer
    - DeviationOfRefer_last.m
    - findbestpoint3.m
    - Trajectory2.m
    - referencni_bez_vyrazovani_pro_doplneni.m
    - referencni_s_vyrazovanim_pro_doplneni.m
    - Vyrazovani

Soubory určené pro spuštění mají před názvem „MAIN“ např. MAIN_KLT.m
Program byl vytvářen v programu Matlab R2015b

B ZIP soubor METODY

- Složka IPAN
    - DeviationOfRefer_last.m
    - FillHole.m
    - FindAllCandidateInitial.m
    - FindAllCandidateLoop.m
    - findbestcandidateInitial.m
    - findbestcandidateLoop.m
    - FindBestPoint.m
    - FindROI.m
    - ChooseBestRow.m
    - MAIN_IPAN.m
    - MainOtherTrajectoryDetect.m
    - MergeTraject.m
    - Pointsuminitial.m
    - Pointsumloop.m
    - resizepoint.m
    - startendtrajectory.m
    - TestHypotesis.m
    - Trajectory2.m
- Složka KLT
    - DeviationOfRefer_last.m
    - findbestpoint3.m
    - MAIN_KLT.m
    - startendtrajectory.m
    - Trajectory2.m
- Složka RC
    - DeviationOfRefer_last.m
    - findbestpoint3.m
    - MAIN_RC.m
    - startendtrajectory.m
    - Trajectory2.m
    - Evaluate_P.m
    - Evaluate_Energy.m
    - findROIActualTemplate.m
    - findROIoldTemplate.m
    - genMesh.m

Soubory určené pro spuštění mají před názvem „MAIN“ např. MAIN_KLT.m
Program byl vytvářen v programu Matlab R2015b