

МИНІСТЭРСТВА АДУКАЦЫІ РЭСПУБЛІКІ БЕЛАРУСЬ

БЕЛАРУСКІ ДЗЯРЖАЎНЫ УНІВЕРСІТЭТ

Факультэт прыкладной матэматыкі і інфарматыкі

Кафедра дыскрэтнай матэматыкі і алгарытмікі

**РЭКАНСТРУКЦЫЯ ПАВЕРХНІ ПА ДАДЗЕНЫХ З
БЕСПЛОТНЫХ ЛЯТАЛЬНЫХ АПАРАТАЎ**

Курсавая праца

Богдана Уладзіслава Уладзіміравіча
студэнта 3 курса
спецыяльнасць "інфарматыка"

Навуковы кіраўнік:
Тузікаў Аляксандр Васільевіч
ген. дырэктар АІПІ НАН Беларусі,
прафесар кафедры ДМА,
доктар фіз.-мат. навук

Мінск, 2017

Анатацыя

У курсавой працы апісаныя этапы пабудовы трохмернай мадэлі па дадзеных з БПЛА, падрабязна разгледжаныя спосабы вымання ключавых кропак на выявах і методы іх апісання, параўнаныя розныя тыпы дэтэктараў і дэскрыптараў, спосабы пошука адпаведнасцяў паміж кропкамі. У працы прадстаўленыя вынікі эксперыменту над выманнем ключавых кропак рознымі алгарытмамі, прыведзенае апісанне праграмнага забеспечэння, над якім вялася распрацоўка, і якое з'яўляецца канечным інструментам пабудовы трохмернай мадэлі.

Аннотация

В курсовой работе описаны этапы построения трёхмерной модели по данным с БПЛА, подробно рассмотрены способы извлечения ключевых точек на изображениях и методы их описания, сравнены разные типы детекторов и дескрипторов, способы поиска соответствий между точками. В работе представлены результаты экспериментов над извлечением ключевых точек разными алгоритмами, приведено описание программного обеспечения, над которым велась разработка, и которое является конечным инструментом построение трёхмерной модели.

Annotation

The stages of the problem of three-dimensional surface reconstruction based on data from UAV (unmanned aerial vehicle) are considered. The ways of detecting, extracting and describing keypoints on the image are described in detail. The series of experiments determining the differences in ways of extracting the keypoints were held. The progress of implementing the software for surface reconstruction from the set of images is described.

ЗМЕСТ

УВОДЗІНЫ	4
ТЭАРЭТЫЧНАЕ АБГРУНТАВАННЕ	5
Простыя метады і метады, заснаваныя на ключавых кропках	5
Ключавыя кропкі і дэскрыптары	6
Апісанні дэтэктараў і дэскрыптараў	6
Спалучэнне простых метадаў і метадаў, заснаваных на ключавых кропках	8
Пошук адпаведнасцяў	8
ПРАКТЫЧНАЯ РЭАЛІЗАЦЫЯ	10
Эксперыменты над асаблівасцямі выяваў	10
Распрацоўка праграмнага забеспячэння для рэканструкцыі паверхні	17
ВЫСНОВЫ	21
СПІС КРЫНІЦАЎ	22

УВОДЗІНЫ

Рэканструкцыя паверхні зямлі па зыходных дадзеных з беспілотных лятальных апаратуў (БПЛА) - працэс, які знайшоў практычнае прымяненне ў вялікай колькасці прыкладных задачаў. Такая задача можа ўзнінуць у працэсе будоўлі, пры ліквідацыі наступстваў кліматычных катастроф альбо ў сельскай гаспадарцы. Развіццю і ўдасканаленню працэса рэканструкцыі паверхні паспрыяла адсутнасць у неабходнасці дарагой і спецыялізаванай тэхнікі: для запуску алгарытмаў рэканструкцыі можа хапіць зыходных дадзеных у выглядзе набора выяваў, зробленых звычайнай камерай на БПЛА якія, у сваю чаргу, з кожным годам становяцца усё больш даступнымі шырокаму карыстачу.

Алгарытмы рэканструкцыі паверхні ўключаюць у сябе вялікую колькасць этапаў, у выніку выканання якіх з набора выяваў і, магчыма, нейкіх дадатковых дадзеных, мы атрымліваем гатовую трохмерную мадэль паверхні, якая можа выкарыстоўвацца ў прыкладных мэтах.

У гэтай працы мы падрабязна разгледзім некаторыя з этапаў пабудовы мадэлі, прывядзэм вынікі эксперыменту з алгарытмамі вымання і апісання ключавых кропак, апішам распрацаваную праграму для пабудовы і візуалізацыі трохмернай мадэлі па наборы выяваў (напрыклад, па дадзеных з БПЛА).

ГЛАВА 1.

ТЭАРЭТЫЧНАЕ АБГРУНТАВАННЕ

У агульным выпадку задачу рэканструкцыі можна падзяліць на некалькі этапаў наступным чынам:

- Апрацоўка зыходных выяваў, пошук адпаведнасцяў паміж імі.
Апрацоўка дадатковых зыходных дадзеных пры іх наяўнасці (калібровачныя параметры камеры, gps-трэк з БПЛА, інш.)
- Запуск алгарытма пошука размяшчэння камераў у прасторы.
Пабудова разрэджанага воблака кропак праз мінімізацыю памылкі (фотаметрычны, праекцыі ці інш.)
- Ушчыльненне воблака кропак, нанясенне тэкстураў для атрымання сапраўднай шчыльной трохмернай мадэлі паверхні зямлі альбо іншага аб'екта

Спынімся падрабязней на першым этапе.

1.1 Простыя метады і метады, заснаваныя на ключавых кропках

Метады, якія даюць яўленне пра пазіцыю камеры ў прасторы і будуюць першасную структуру сцэны, могуць быць падзеленыя на два класы. Разгледзім іх падрабязней.

Першы - метады заснаваныя на *пошуку асаблівасцяў* (англ. *feature-based methods*). Ідэя у выманні з кожнай асобнай выявы пэўнай колькасці асаблівасцяў (імі могуць быць, напрыклад, кропкі альбо лініі) і пошук адпаведнасцяў паміж асаблівымі кропкамі на розных выявах (англ. *matching*), аднаўленне пазіцыяў камеры і структуры сцэны з дапамогай эпіпалярнай геаметрыі (геаметрыя стэрэабачання) і, у завяршэнне, удакладненне параметраў праз мінімізацыю памылкі праекцыі. Падобную працэдуру выконвае вялікая колькасць алгарытмаў: даступнасць эфектыўных метадаў вымання асаблівасцяў і пошуку іх адпаведнасцяў дазваляе рабіць гэта хутка і надзейна; разам з тым, такі падыход можа даваць недастатковую дакладнасць вынікаў, быць прымяняльным толькі на вызначаным класе зыходных дадзеных (напрыклад, не дапускаць аднастайныя тэкстуры), патрабуе надзейных алгарытмаў ацэнкі пазіцыі.

Другі клас метадаў заснаваны на гэтак званых *простых метадах* (англ. *direct methods*): у такіх метадах непасрэдна выкарыстоўваюцца значэнні запісаныя ў пікселях, не адбываецца спробы выяўлення на выяве вызначанага кшталту асаблівасцяў. Для апрацоўкі выкарыстоўваецца

накірунак і велічыні лакальнага градыента інтэнсіўнасці. Простыя метады, якія выкарыстоўваюць усю інфармацыю на выяве, нават у зонах з маленькім градыентам, апынуліся значна эфектыўнейшымі за метады, заснаваныя на пошуку асаблівасцяў у сцэнах з нізкай тэкстураванасцю альбо ў выпадку размытасці альбо дэфакусіроўкі [1]. Дастаткова праца затратнай працэдурай становіща падлік фотаметрычнай памылкі (у параўнанні з падлікам памылкі практыкі). Разам з тым, паколькі праца вядзеца непасрэдна са значэннямі пікселяў, час на пошук асаблівасцяў і падлік дэскрыптараў можа быць захаваны.

1.2 Ключавыя кропкі і дэскрыптары

Тут і далей мы будзем казаць пра “ключавыя кропкі” (англ. *keypoints*), хаця ў тэрмінах таго ці іншага дэтэктара ключавой кропкай можа называцца любая адзінка інфармацыі: кропка, акружнасць, лінія, кропка з накірункам і інш.

Дэтэктарам (англ. *detector*) будзем называць алгарытм, які знаходзіць на выяве ключавыя кропкі.

Знойдзеная ключавая кропка пасля звычайна апісваецца ў нейкай вызначанай кароткай форме. Такое апісанне ключавой кропкі будзем называць дэскрыптарам (англ. *descriptor*). Часцей за ўсё сустракаюцца дэскрыптары ў выглядзе рэчаісных вектароў альбо бінарных радкоў. Дэскрыптары дапамагаюць хутка знаходзіць адпаведныя адна адной ключавыя кропкі на розных выявах. Знайсці адлегласць паміж двумя дэскрыптарамі і, такім чынам, вызначыць іхняе падабенства, з'яўляеца асноўнай аперацыяй пры пошуку адпаведнасцяў паміж двумя выявамі, таму да надзейнасці і хуткасці падліку функцыі адлегласці прад'яўляюцца асаблівасці патрабаванні. На практыцы адлегласць часцей за ўсё падліваецца праз Эўклідаву адлегласць L2 (для рэчаісных вектароў) альбо праз адлегласць Хэмінга (для бінарных радкоў).

Ніжэй прывядзем кароткае апісанне найбольш распаўсюджаных дэтэктараў ключавых кропак і спосабаў іх апісання.

1.3 Апісанні дэтэктараў і дэскрыптараў

Дэтэктар **FAST** (Features from Accelerated Segment Test) [2]. Як вынікае з назвы, асноўнай перавагай дэтэктара з'яўляеца ягоная хуткасць, гэта асабліва важна ў выпадку, калі дадзеныя паступаюць і патрабуюць апрацоўкі ў рэжыме рэальнага часу і абмежаваных рэурсаў. Найлепшым прыкладам будуць SLAM-прыкладанні (*Simultaneous Localization and Mapping*), у якіх у рэжыме рэальнага часу адбываецца

ацэнка месца заходжання і пошук яго на мапе; SLAM-прыкладанні часта заходзяць прымяненне на мабільных прыладах, у тым ліку на БПЛА.

FAST з'яўляецца алгарытмам “пошуку вуглоў” (англ. *corner detector*): кропка з'яўляецца ключавой, калі ў маленькіх выколіцах пікселя алгарыту атрымліваецца распознаць шаблон, іншымі словамі “вугал”. Дадатковая аптымізацыя адбываецца з дапамогай метадаў машыннага навучання.

Даследванні паказваюць, што ён у некалькі разоў хутчэйшы за іншыя існуючыя алгарытмы пошука ключавых кропак на аснове вылучэння вуглоў, але, ў сваю чаргу ён недастаткова ўстойлівы да моцна зашумленых выявав.

Дэскрыптар і дэтэктар **SIFT** [3] быў прадстаўлены ў 2004 годзе і сёння з'яўляецца адным з найбольш распаўслюджаных. Важнымі харектарыстыкамі алгарытмамі з'яўляецца ўстойлівасць да масштабавання і змянення арыентацыі выявы, яе афінных пераўтварэнняў і зашумленых варыянтаў. Яскрава выраджаная харектэрнасць ключавых кропак дазваляе паспяхова шукаць адпаведнасці паміж рознымі выявамі. У [3], апроч апісання спосаба вымання ключавых кропак і фармата дэскрыптара, апісваюцца тэхнікі для эфектыўнага пошука адпаведнасцяў паміж выявамі з SIFT-дэскрыптарамі, а таксама алгарытмы эфектыўнага распознавання патэрнаў. Такая завершанасць даследавання алгарыту дае яму вялікую перавагу перад іншымі і моцна паспрыяла ягонаму распаўсюду. SIFT-дэскрыптарам з'яўляецца рэчаісны вектар размернасці 128.

Дэскрыптар і дэтэктар **SURF** [4] быў упершыню апубліканы ў 2006 годзе і пачаткова ідэя стварэння заключалася ў стварэнне “больш хуткага” SIFT-а. Ён таксама з'яўляецца ўстойлівым да змены масштаба і паварота, можа быць паспяхова прыменены для размытых выявав; разам з тым, выкарыстанне SURF-а абмежаванае, калі мы маем справу са зменамі ў вугле назірання. Дэскрыптар SURF можа быць як размернасці 64, так і 128 (як SIFT).

Звернем увагу на тое, што і SIFT і SURF з'яўляюцца запатэнтаванымі алгарытмамі і, негледзячы на вялікую колькасць рэалізацыяў з адкрытым зыходным кодам, іх выкарыстанне ў камерцыйных прадуктах абмежаванае. У сваю чаргу, алгарытм ORB прарыетарным не з'яўляецца.

Дэскрыптар і дэтэктар **ORB** (Oriented FAST and Rotated BRIEF) [5] з'яўляецца своеасаблівой камбінацыяй ключавых кропак FAST і дэскрыптара BRIEF. Мэтай стварэння ORB было пераўзысці SIFT у хуткасці (нават ахвяруючы надзеінасцю і дакладнасцю) у сувязі з распаўсюдам мабільных маламагутных прыладаў. Дэскрыптары маюць выгляд бінарных радкоў. Адпаведнасці паміж ORB дэскрыптарамі эфектыўна могуць быць знайдзеныя з дапамогай LSH (Locality-sensitive

hashing).

Дэскрыптар **BRIEF** (Binary Robust Independent Elementary Features) [6] стаў адным з першых дэскрыптараў, якія апісываюць ключавыя кропкі з дапамогай бінарных радкоў (SIFT альбо SURF, напрыклад, выкарыстоўваюць вектары з рэчаіснымі лікамі якія, у большасці выпадкаў, пасля ўсё адно ў мэтах эфектыўнасці фарматуюцца ў бінарныя радкі, што дазваляе выкарыстоўваць эфектыўную адлегласць Хэмінга).

Дэскрыптар і дэтэктар **BRISK** (Binary Robust Invariant Scalable Keypoints) [7] ствараўся як альтэрнатыва вышэйзгаданым SIFT і SURF, хуткасць працы якога дасягаецца праз выкарыстанне новага, заснаванага на FAST, дэтэктара ў камбінацыі з бінарным дэскрыптарам.

Звернем увагу на тое, што некаторыя алгарытмы спалучаюць у сабе адразу як пошук ключавых кропак, так і падлік іх дэскрыптараў. У агульным выпадку гэты працэс можа быць яўна падзелены: на мностве ключавых кропак атрыманых любым з дэтэктараў можа быць запушчаны алгарытм, які для кожнай кропцы паставіць у адпаведнасць дэскрыптар вызначанага фармату. На практицы, некаторыя пары дэтэктар + дэскрыптар спалучаюцца добра і заўжды выкарыстоўваюцца разам, некаторыя сумяшчаюцца вельмі кепска. Большасць алгарытмаў, якія знайшлі шырокое прыменение, апісываюць адразу спосаб дэтэкцыі і апісання і ўтвараюць такім чынам закрытую сістэму, раздзяленне якой не прыводзіць да добрых вынікаў.

1.4 Спалучэнне простых метадаў і метадаў, заснаваных на ключавых кропках

Вядомыя эфектыўныя алгарытмы, якія спалучаюць у сабе простыя метады і метады, заснаваныя на выманні асаблівасцяў. Найбольшы распаўсюд такія алгарытмы знайшлі ў задачах апрацоўкі плыні дадзеных (відэаплыня, апрацоўка дадзеных у рэальнym часе з БПЛА), калі алгарытм будзе мапу мясцовасці і шукае сваё месцазнаходжанне ў рэальнym часе.

Напрыклад, у [8] яўны пошук асаблівасцяў выклікаецца толькі калі новы кадр запускае ініцыялізацыю новых 3D кропак у прасторы, іначай выкарыстоўваюцца простыя метады якія, у сваю чаргу, няяўна даюць адпаведнасці паміж кропкамі, пазначанымі раней у якасці асаблівых.

1.5 Пошук адпаведнасцяў (matching)

Пошук асаблівых кропак і падлік адпаведных дэскрыптараў з'яўляецца, вядома ж, толькі падрыхтоўчым этапам да этапа параўнання дэскрыптараў з дзвюх выявав і пошук найлепшых параў (пратэс

матчынга, англ. *matching*). Не існуе ўніверсальна спосаба, які даваў бы найлепшыя вынікі на любым тыпе асаблівых кропак і іх дэскрыптараў: для кожнага набора дэскрыптараў існуе найбольш эфектыўны спосаб пошуку адпаведнасцяў. Таксама, выбар матчара залежыць ад пастаўленых намі задачаў і расстаўленых прыарытэтаў: ці нам найважнейшая хуткасць, ці дакладнасць, ці дастаткова нам толькі n найлепшых супадзенняў ці мы хочам атрымаць усе знайдзеныя. Важным пытаннем з'яўляецца вызначэння парога, калі знайдзеная пара дэскрыптараў з'яўляецца заведама хібнай.

Самым простым спосабам знайсці адпаведнасці паміж дэскрыптарамі з дзвюх розных выяваў з'яўляецца прости пералік усіх магчымых пароў, падлік нормы (L_2 , Хэмінга, альбо любой іншай) і сартыроўка ўсіх пароў па ўзрастанні значэнняў адлегласцяў (гэтак званы Brute-Force Matcher). Праз прастату такі спосаб з'яўляецца простым у рэалізацыі, беспамылковым, але вельмі марудным. Практычнае прымяне на ўмовах апрацоўкі плыняў дадзеных альбо на пост-апрацоўцы вялікіх наборах дадзеных, немагчымае.

Для выпадкаў, калі дакладнасцю можна ахвяраваць на карысць хуткасці (напрыклад, у працы з БПЛА хуткасць апрацоўкі грае першасную ролю), былі вынайдзеныя і іншыя алгарытмы пошуку адпаведнасцяў.

Метады, пра якія ідзе гаворка, маюць агульную назvu *метадаў набліжанага пошуку бліжэйшых суседзяў* (англ. *Approximate Nearest Neighbors*). Падлічаныя намі дэскрыптары з'яўляюцца звычайнім мноствам вектороў, на якім зададзеныя суадносіны адлегласці. У практычнай частцы мы будзем выкарыстоўваць два віда такіх метадаў. Для дэскрыптараў з рэчаіснымі вектарамі (SIFT, SURF) гэта будзе метад заснаваны на k -мерных дрэвах (англ. *k -dimensional trees*), для бінарных дэскрыптараў - Locality-sensitive Hashing (LSH) - імавернасны метад паніжэння размернасці дадзеных.

ГЛАВА 2.

ПРАКТЫЧНАЯ РЭАЛІЗАЦЫЯ

У гэтым раздзеле я хачу апісаць дзве практычныя часткі, зробленыя мной. Першая - серыя эксперыментаў над выманнем асаблівых кропак, падлікам іх дэскрыптараў і пошукам адпаведнасцяў паміж дзвюма выявамі з аднаго набору. Другая - апісанне распрацаванага праграмнага забеспячэння, якое выконвае ўесь цыкл рэканструкцыі, пабудаванае на аснове бібліятэкі [9].

2.1 Эксперыменты над спосабамі вымання асаблівых кропак, іх апісання і пошуку

Практычна-даследніцкая частка маёй працы - парайнанне розных дэтэктараў, дэскрыптараў, тыпаў ключавых кропак на рознага кшталту дадзеных, пошук узаемасувязі паміж тыпам пачатковых дадзеных і якасцю вынікаў, у залежнасці ад прымененых алгарытмаў.

Для правядзення эксперыменту я выбраў 4 набора дадзеных па 2 выявы. Крытэрам выбара была разнастайнасць: выявы павінны быць моцна альбо слаба тэкстураванымі; адрознівацца паваротам, паралельным пераносам альбо выбарам іншага вугла камеры і г.д., яскравайвыраджанай альбо аднастайнай каліяровай гамай. Далей ідзе кароткае апісанне кожнага з набораў.

- *church*: два фотаздымкі царквы ў Швейцарыі, зробленыя з БПЛА. Маюць яскравую выраджанасць асобных аб'ектаў, у першую чаргу будынкаў. Гл. малюнкі 1.
- *fountain*: два фотаздымкі фантана, зробленыя пад розным вуглом. Пераважвае адзіная каліяровая гамма, выявы слаба тэкстураваныя. Гл. малюнак 2.
- *ellis*: два фотаздымкі з выглядам на Манхэтэн, прыблізна палову кожнага здымка займае аднастайныя тэкстуры: вада, дарога. Аб'екты, якія знаходзяцца на абедвух выявах, моцна адрозніваюцца масштабам. Гл. малюнак 3.
- *grass*: два фотаздымкі беларускай сельскай мясцовасці зробленыя з БПЛА. Большая частка абедвух выяваў занятыя аднастайнай, практычна алнакаліяровай тэкстурой. Гл. малюнак 4

Усе выявы былі папярэдне прыведзеныя да парайональных памераў: вышыня кожнай выявы склада 640 пікселяў, шырыня вылічвалася з захаваннем прапорцыяў.

Усе эксперыменты праводзіліся пры аднолькавых умовах, на камп'ютары з усталяванай ubuntu 16.04 з 2gb RAM на аднаядравым працэсары. Рэалізацыі алгарытмаў браліся з бібліятэкі камп'ютарнага зроку з адкрытым зыходным кодам OpenCV.

Для кожнага набора дадзеных падлікі прадстаўлены ў дзвюх табліцах. У першай табліцы ідзе апісанне часавых характеристыкаў дэтэкцыі крапак і падліку іх апісання рознымі алгарытмамі. У другой - часавыя і колькасныя характеристыкі алгарытмаў пошуку адпаведнасцяў паміж дзвюма выявамі з набора.

Вынікі эксперымантаў для дадзеных *church* прыведзены ў табліцах 1 і 2, *fountain* - 3 і 4, *ellis* - 5 і 6, *grass* - 7 і 8. У кожнай табліцы час прыведзены ў секундах.



Малюнак 1: набор дадзеных *church*

Тып дэтэктара	Сярэдняя колькасць крапак	Час дэтэкцыі	Час на адну ключавую крапку	Тып дэскрыптара	Час на падлік дэскрыптараў	Час на падлік аднаго дэскрыптара
SIFT	4109	0.3951	0.0000481	SIFT	0.6190	0.0000753
SURF	5291	0.4141	0.0000391	SURF	1.3550	0.0001280
BRISK	13478	0.3624	0.0000134	BRISK	0.2292	0.0000085
ORB	3000	0.0359	0.0000060	ORB	0.0213	0.0000036
AKAZE	3655	0.4810	0.0000658	AKAZE	0.4024	0.0000550
CenSurE	1077	0.0240	0.0000111	BRIEF	0.0070	0.0000032

Табліца 1: пошук асаблівых крапак і падлік адпаведных дэскрыптараў на наборы *church*

Дэтэктар, дэскрыптар	BF адпаведнасцяў	BF час	BF-knn адпаведнасцяў	BF-knn час	FLANN адпаведнасцяў	FLANN час
SIFT	2207	1.5363	83	1.5089	62	0.1324
SURF	2438	1.1988	118	1.1760	67	0.1266
BRISK	6569	6.9033	89	6.9348	45	1.0028
ORB	1575	0.2020	22	0.1912	19	0.0474
AKAZE	1936	0.4880	145	0.5027	93	0.2132
CenSurE+ BRIEF	509	0.0230	38	0.0233	19	0.0100

Табліца 2: пошук адпаведнасцяў на наборы *church*



Малюнак 2: набор дадзеных *fountain*

Тып дэтэктара	Сярэдняя колькасць кропак	Час дэтэкцыі	Час на адну ключавую кропку	Тып дэскрыптара	Час на падлік дэскрыптараў	Час на падлік аднаго дэскрыптара
SIFT	2953	0.4213	0.0000713	SIFT	0.5333	0.0000903
SURF	4703	0.4138	0.0000440	SURF	1.4973	0.0001592
BRISK	2142	0.0583	0.0000136	BRISK	0.0387	0.0000090
ORB	2907	0.0299	0.0000051	ORB	0.0248	0.0000043
AKAZE	776	0.3693	0.0002378	AKAZE	0.2518	0.0001622
CenSurE	173	0.0316	0.0000913	BRIEF	0.0038	0.0000110

Табліца 3: пошук асаблівых кропак і падлік адпаведных дэскрыптараў на наборы *fountain*

Дэтэктар, дэскрыптар	BF адпаведнасцяў	BF час	BF-knn адпаведнасцяў	BF-knn час	FLANN адпаведнасцяў	FLANN час
SIFT	1399	0.7398	68	0.7324	40	0.0827
SURF	2082	0.9625	70	0.9743	44	0.1080
BRISK	974	0.1673	28	0.1581	22	0.0365
ORB	1446	0.2204	18	0.2039	11	0.0660
AKAZE	338	0.0210	35	0.0213	25	0.0083
CenSurE+ BRIEF	63	0.0008	14	0.0007	15	0.0009

Табліца 4: пошук адпаведнасцяў на наборы *fountain*



Малюнак 3: набор дадзеных *ellis*

Тып дэтэктара	Сярэдняя колькасць крапак	Час дэтэкцыі	Час на адну ключавую крошку	Тып дэскрыптара	Час на падлік дэскрыптараў	Час на падлік аднаго дэскрыптара
SIFT	2725	0.4967	0.0000911	SIFT	0.6985	0.0001281
SURF	4950	0.4373	0.0000442	SURF	1.3399	0.0001353
BRISK	3619	0.0883	0.0000122	BRISK	0.0716	0.0000099
ORB	2947	0.0217	0.0000037	ORB	0.0240	0.0000041
AKAZE	917	0.3723	0.0002029	AKAZE	0.2570	0.0001401
CenSurE	261	0.0311	0.0000597	BRIEF	0.0023	0.0000044

Табліца 5: пошук асаблівых крапак і падлік адпаведных дэскрыптараў на наборы *ellis*

Дэтэктар, дэскрыптар	BF адпаведнасцяў	BF час	BF-knn адпаведнасцяў	BF-knn час	FLANN адпаведнасцяў	FLANN час
SIFT	1265	0.8107	158	0.6754	142	0.0785
SURF	1746	1.0627	149	1.0799	118	0.1052
BRISK	1796	0.6418	93	0.6134	78	0.1154
ORB	1469	0.1894	111	0.2477	92	0.0463
AKAZE	494	0.0343	76	0.0368	58	0.0142
CenSurE+ BRIEF	126	0.0017	9	0.0020	11	0.0015

Табліца 6: пошук адпаведнасцяў на наборы *ellis*



Малюнак 4: набор дадзеных *grass*

Тып дэтэктара	Сярэдняя колькасць кропак	Час дэтэкцыі	Час на адну ключавую кропку	Тып дэскрыптара	Час на падлік дэскрыптараў	Час на падлік аднаго дэскрыптара
SIFT	5357	0.6637	0.0000619	SIFT	0.9163	0.0000855
SURF	15894	1.1313	0.0000356	SURF	3.5399	0.0001114
BRISK	46305	1.3892	0.0000150	BRISK	0.8116	0.0000088
ORB	3000	0.0921	0.0000154	ORB	0.0313	0.0000052
AKAZE	3057	0.6880	0.0001125	AKAZE	0.7078	0.0001158
CenSurE	829	0.1565	0.0000944	BRIEF	0.0199	0.0000120

Табліца 7: пошук асаблівых кропак і падлік адпаведных дэскрыптараў на наборы *grass*

Дэтэктар, дэскрыптар	BF адпаведнасцяў	BF час	BF-knn адпаведнасцяў	BF-knn час	FLANN адпаведнасцяў	FLANN час
SIFT	3043	2.6474	520	2.5827	452	0.1635
SURF	8704	11.6172	848	11.5856	712	0.4455
BRISK	24512	93.8451	1852	94.3301	1460	9.8551
ORB	1624	0.2333	238	0.4267	207	0.1301
AKAZE	1735	0.4762	732	0.4167	668	0.4133
CenSurE+ BRIEF	452	0.0177	181	0.0188	172	0.0108

Табліца 8: пошук адпаведнасцяў на наборы *grass*

Кожны з набораў дадзеных апрацоўваўся шасцю рознымі алгарытмамі пошуку і падліку ключавых кропак. Пры правядзенні эксперымента была спроба прымянення і іншых алгарытмаў, якія не прысутнічаюць цяпер у табліцах, аднак яны не паказалі вынікаў парадунальных з вышэй прыведзенымі. Напрыклад, натуральний падавалася пара дэтэктара FAST і дэскрыптар ORB, якая, аднак, паказала кепскія вынікі як па часе, так і па якасці знайдзеных адпаведнасцяў.

Прывядзем апісанне ўжытых метадаў пошука адпаведнасцяў згодна з іх скарачэннямі ў табліцах:

- *BF* - звычайны BruteForce алгарытм, реалізацыя з OpenCV, запускаўся з параметрам *CrossCheck = True*: гэты параметр кажа алгарытму вяртаць пару спалучаных дэскрыптараў толькі тады, калі кожны з дэскрыптараў на чарговым кроку выступае найлепшым для свайго адпаведніка на іншай выяве (то бок дэскрыптары павінны спалучацца “у абодва бакі”). Гэта з'яўляецца своеасаблівай альтэрнатывай “тэсту суадносінаў” (англ. *ratio test*), апісаному ў [3] - гэты тэст мы будзем запускаць разам з наступным метадам.
- *BF-knn* - BruteForce алгарытм, які структуруе знайдзенія адпаведнасці крыху іншым чынам, чым вышэйзгаданы BF: для кожнага дэскрыптара ён знаходзіць не больш за $k = 2$ адпаведных дэскрыптараў. Далей мы запускаем “тэст суадносінаў”, які апісаны ў

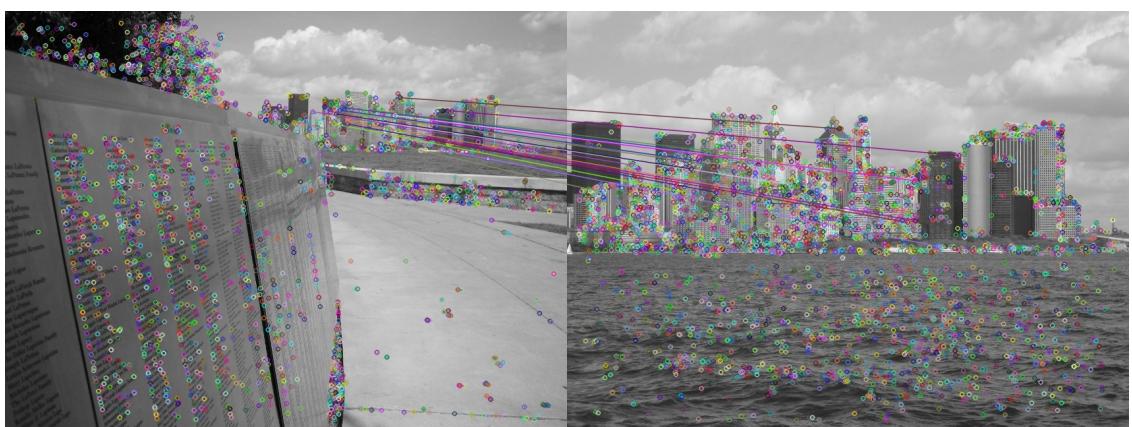
[3] (старонка 20). Сцвярджаеща, што калі адлегласці двух найлепшых знайдзеных адпаведнікаў для вызначанага дэскрыптара адпавядаюць няроўнасці $distance_1 < ratio \times distance_2$, дзе $ratio$ - канстанта, якая звычайна абіраецца з прамежка $[0.7, 0.8]$, то першы адпаведнік для знайдзенага дэскрыптара з'яўляеца “дастаткова добрым” - такія дэскрыптары заносяцца ў табліцу як “знайдзеныя”. Трэба заўважыць, што гэты “тэст суадносінаў” на практыцы праходзіць дастаткова мала параў дэскрыптараў.

- *FLANN* - сямейства алгарытмаў набліжанага пошуку бліжэйшых суседзяў. У нашым выпадку, для SIFT і SURF гэты алгарытм заснаваны на KD-дрэвах, для астатніх дэскрыптараў гэта LSH (Locality-sensitive hashing).

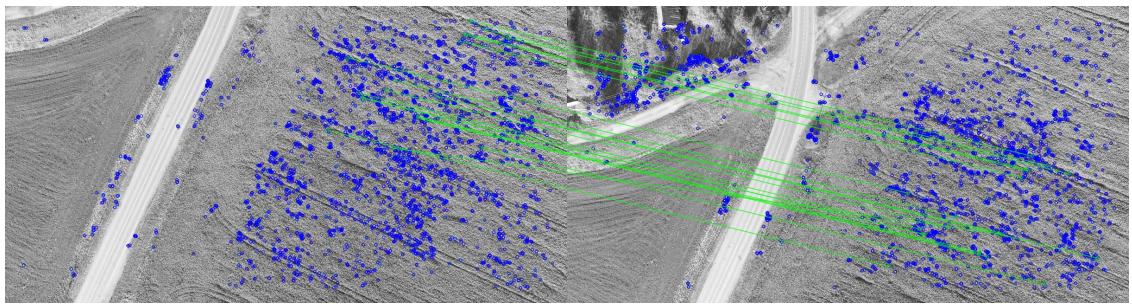
Заўвагі:

- алгарытм ORB прымае ў якасці аднаго з параметраў абмежаванне зверху на колькасць знайдзеных ключавых крапак; у табліцах 1 - 8, калі колькасць ключавых крапак, знайдзеных ORB, роўная 3000, гэта не азначае, што гэта максімальна магчымы лік; для астатніх алгарытмаў прыведзеныя максімальныя значэнні.

На малюнках 5 - 6 прыведзеныя два прыклады знайдзеных адпаведнасцяў. Колькасць нанесеных у выглядзе лініяў адпаведнасцяў абмежаваная лікам 20 для паляпшэння нагляднасці.



Малюнак 5: знайдзеныя адпаведнасці (ellis, BRISK, BF)



Малюнак 6: знайдзеныя адпаведнасці (grass, ORB, FLANN)

Высновы:

- Алгарытмы дастаткова яскрава падзяліся на “хуткія” і “марудныя”. ORB заўсёды хутчэй за ўсіх знаходзіць ключавыя кропкі і падлічае для іх дэскрыптары (часам гэтаму спрыяе штучнае абмежаванне ў 3000 на колькасць ключавых крапак). Пашук адпаведнасцяў таксама заўсёды адбываецца хутка (у параўнанні з іншым алгарытмамі), хаця і моцна вар’юецца: ад 0.02с да 0.4с.
- Пара алгарытмаў CenSurE+BRIEF знаходзіць малую колькасць ключавых крапак (часам на парадак менш, чым іншыя алгарытмы на тых жа наборах дадзеных), прычым па часе могуць прыйграць таму ж ORB. Негледзячы на гэта, маленькая колькасць дазваляе вельмі хутка знаходзіць адпаведнасці.
- Алгарытм BRISK кепска прымяняецца да аднастайных дадзеных: на наборы *grass* колькасць знайдзеных ключавых крапак рэзка ўзрастает (у 3 разы больш за наступны па колькасці знайдзеных крапак алгарытм). На іншых наборах дадзеных BRISK паводзіць сябе зусім іншым чынам. У выніку алгарытм выкарыстаў рэкордны для нашага даследвання час на пошук асаблівасцяў (94с на BF і BF-knn). Трэба сказаць, што падобныя паводзіны за BRISK-ам былі зауважаныя і на іншых выявах са слабай тэкстураванасцю.
- У агульным выпадку можна казаць пра маруднасць алгарытма SURF на этапе падліку дэскрыптараў: на ўсіх чатырох наборах ён падлічае значэнні дэскрыптараў з самым вялікім абсолютным значэннем па часе, і быў адным з самых марудных у пераліку на час на адзін дэскрыптар.

Агульнае падсумаванне можна сформуляваць наступным чынам: існуюць алгарытмы для хуткай апрацоўкі і для больш грунтоўнай, алгарытмы, якія паказваюць падобныя вынікі на любых дадзеных і алгарытмы, слаба прымяняльныя на канкрэтных дадзеных. SIFT і SURF можна ўмоўна аднесці да надзеіных, але марудных: яны часта

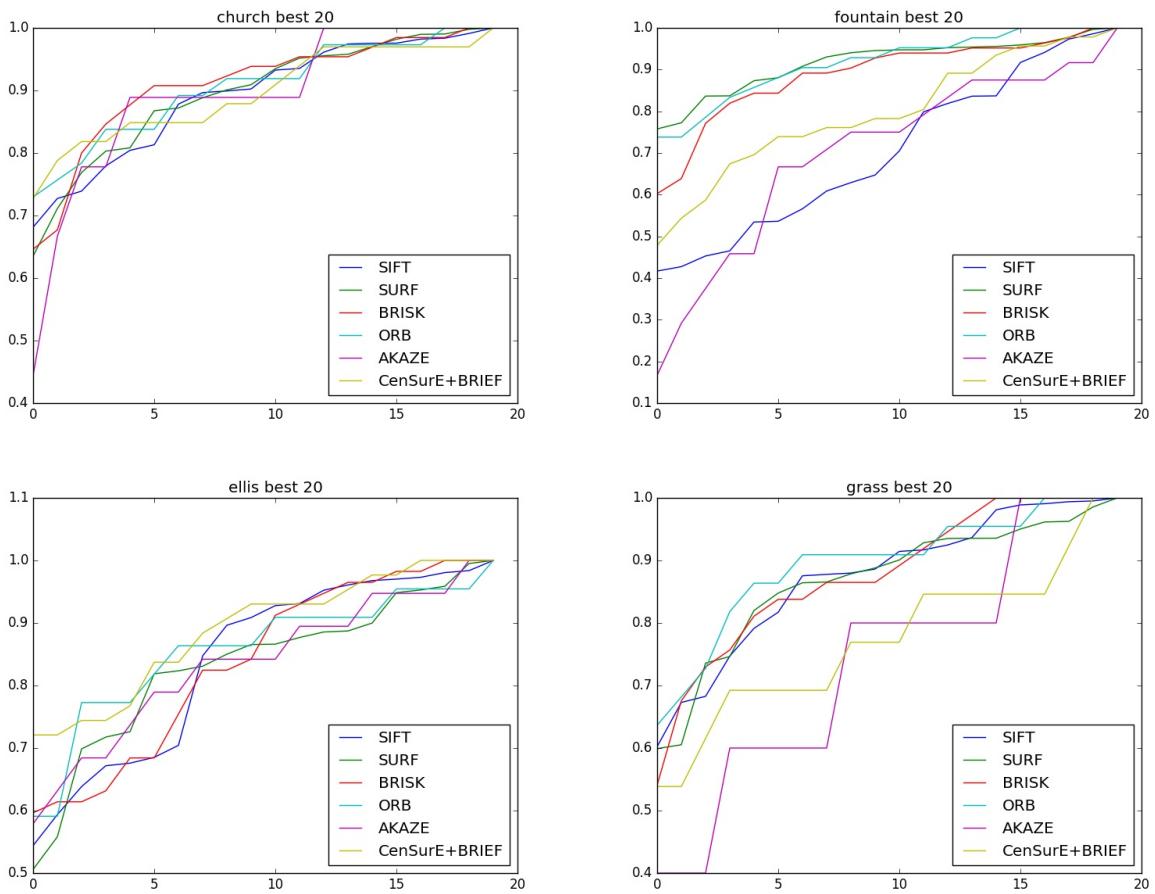
выкарыстоўваюцца на прыладах з добрымі характарыстыкамі, але радзей знаходзяць сваё прымяненне да мабільных прыладах. Нагадаем, што абодва гэтыя алгарытмы апісваюць ключавыя кропкі пры дапамозе рэчаісных вектароў, што, вядома, марудней за бінарныя радкі. Ва ўмовах абмежаваных рэурсаў часта знаходзяць сваё прымяненне алгарытмы ORB і AKAZE: у нашых лічбах яны з'яўляюцца найхутчэйшымі і найменш залежнымі ад рэурсаў. BRISK паводзіць сябе на ўзору іншых алгарытмаў (па колькасных і часавых характарыстыках), але на такіх наборах, як *church* альбо *grass* паказвае рэзкае адставанне па часе на этапе пошуку адпаведнасцяў праз вялікую колькасць папярэдне вылучаных асаблівых кропак.

Цікаўнасць для даследвання прадстаўляе таксама хуткасць роста ад дыстанцыі ў адпаведаных масівах знайдзеных адпаведнасцяў. Дыстанцыя, па сутнасці, з'яўляецца непасрэднай характарыстыкай якасці знайдзенай адпаведнасці, а таксама проста ўплывае на тое, ці пройдзе пара “тэст суадносінаў”, які апісываецца вышэй.

На графіках 7 - 8 прадстаўленыя адлегласці лепшых адпаведна 20 і 250 параў дэскрыптараў. Пары былі суаднесеныя BF алгарытмам. Дадзеная на графіках палічаная адносна апошняй (адпаведна, 20ай альбо 250ай) адлегласці. Інтэрпрэтуюцца дадзеная наступным чынам: рэзкі рост графіка абазначае рэзкі рост кожнай наступнай адлегласці, адпаведна - адлегласць паміж кожнай наступнай парай дэскрыптараў значна больш. Разам з тым, больш спадзісты характар графіка сведчыць пра нізкі рост у значэннях адлегласцяў. На графіках часцей за ўсё асона выбіваецца AKAZE - у табліцах 1 - 8 гэта адлюстроўваецца ў тым, што на некаторых наборах дадзеных суадносіны “колькасць адпаведнасцяў знайдзеных BF-knn да пачатковай колькасці дэскрыптараў” для AKAZE найвялікшая.

2.2 Распрацоўка праграмнага забеспячэння для рэканструкцыі паверхні

Асноўнай практычнай часткай маёй курсавой працы была рэалізацыя канечнай праграмы для рэканструкцыі паверхні па дадзеных з БПЛА. Праграма напісаная на мове C++ з выкарыстаннем фрэймворка Qt на базе бібліятэкі камп'ютарнага зроку з адкрытым зыходным кодам TheiaSfm [9]. Праграма знаходзіцца ў актыўнай распрацоўцы, на дадзены момант рэалізаваная ўсе асноўныя магчымасці, такія як адкрыццё/стварэнне новых праектаў, пабудова разрэджанага воблака кропак па наборы выяваваў, трохмерная візуалізацыя мадэлі, падсветка выбранай камеры, паўторнае выкарыстанне дадзеных (напрыклад, здзейсніўшы адзін раз пошук ключавых кропак, наступны раз гэты этап



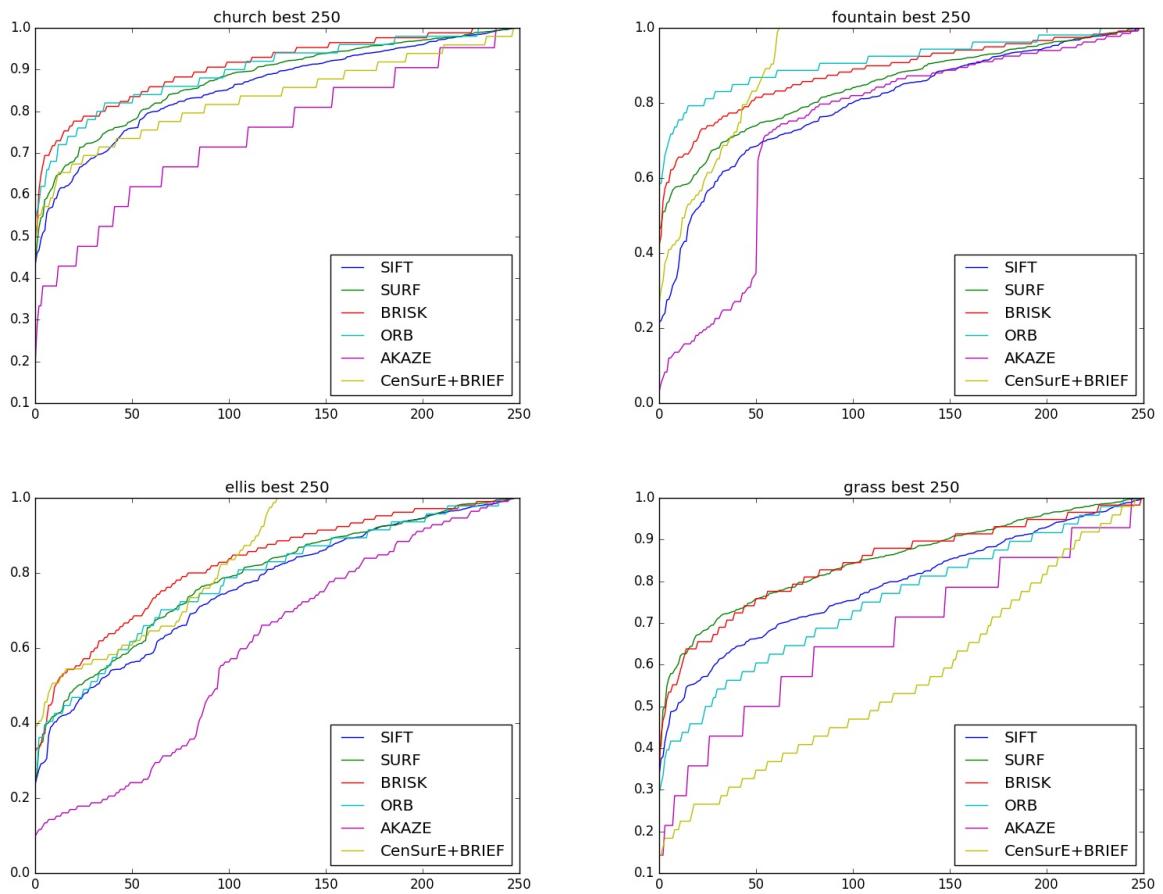
Малюнак 7: рост адлегласці ў 20 лепшых парамах

можа быць прапушчаны, што моцна эканоміць час і рэсурсы), генерацыя справаздачаў пасля рэканструкцыі. Праграма працуе як у кансольным рэжыме, так і ў рэжыме з графічным інтэрфейсам.

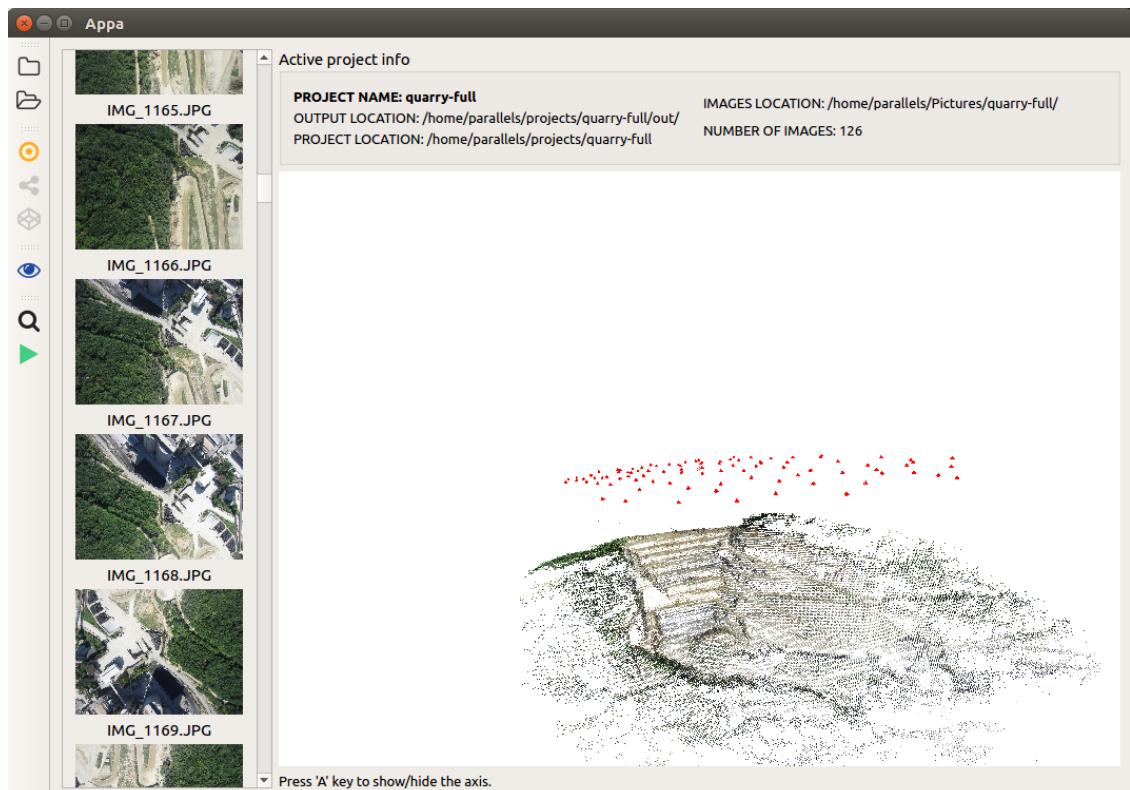
Праграма ствараецца з практичнымі метадамі: яна будзе карысная як для пабудовы мадэлі канечным карыстачом (выкарыстоўваючы графічны інтэрфейс), так і, напрыклад, для правядзення эксперыментуў з алгарытмамі (у гэтым выпадку карыснай будзе магчымасць выклікаць уесь функцыонал праграмы з кансольнага радка).

Праграма кросплатформенная, праца ўсяго функцыяналу была пратэставаная на аперацыйных сістэмах macOS і Ubuntu.

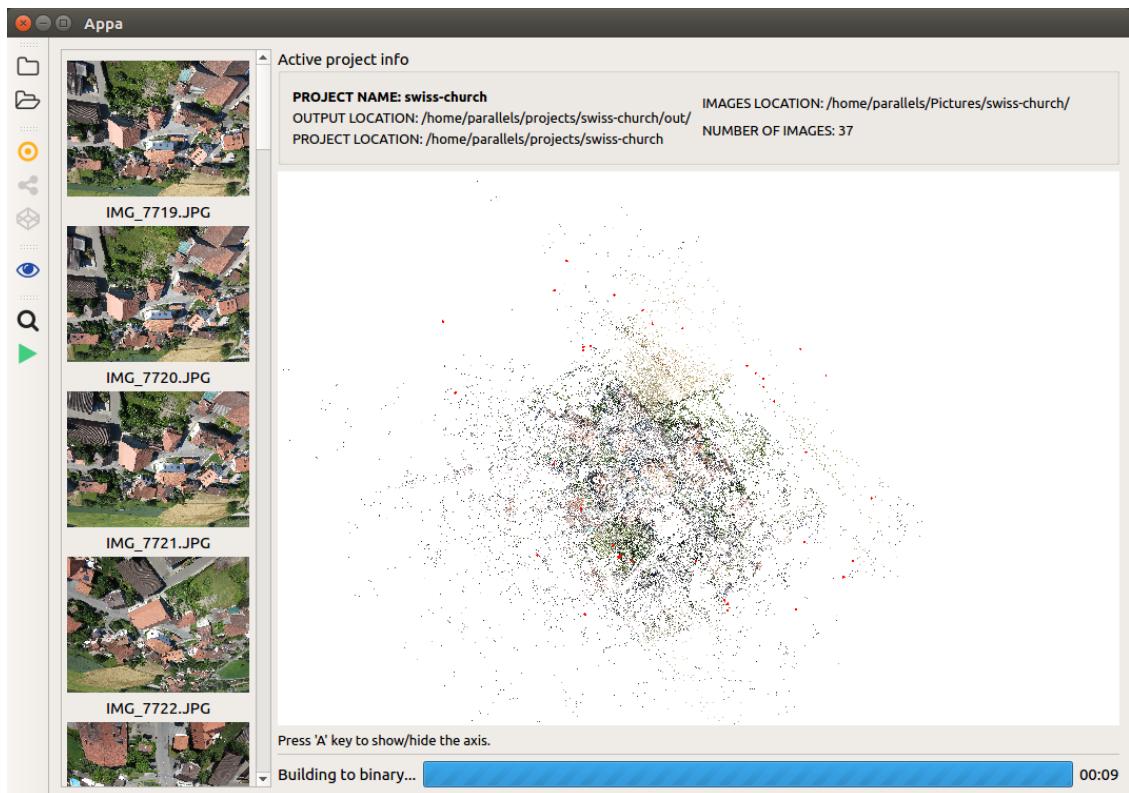
Здымкі экрана з запушчаным прыкладаннем прадстаўленыя на малюнках 9 - 11.



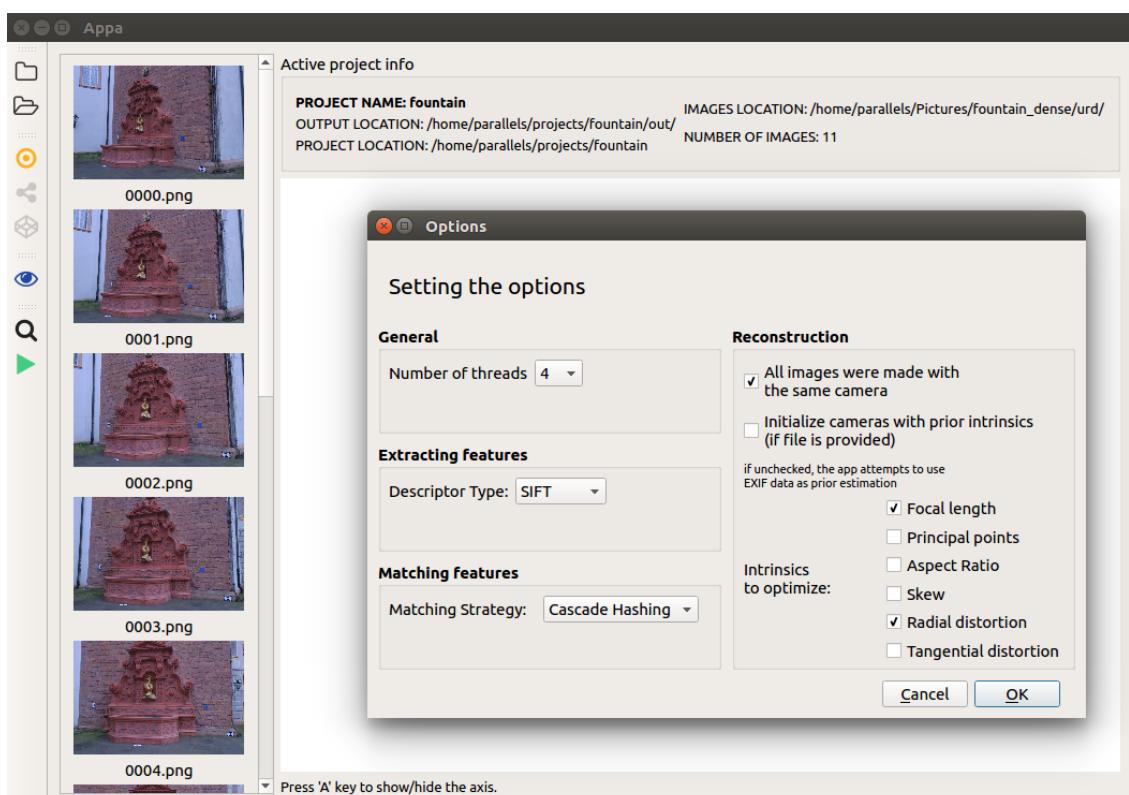
Малюнак 8: рост адлегласці ў 250 лепшых парах



Малюнак 9: здымак экрана



Маlionак 10: здымак экрана



Маlionак 11: здымак экрана

Увесь зыходны код адкрыты і размешчаны на GitHub.

ВЫСНОВЫ

Мною была разгледжаная і засвоеная агульная тэорыя пабудовы трохмернай рэканструкцыі, а таксама шматлікія тэарэтычныя аспекты задачаў, якія сустракаюцца ў галіне камп'ютарнага зроку. Пры засваенні тэорыі я падрабязна спыніўся на этапе пошуку адпаведнасцяў паміж некалькімі выявамі і на этапе, які яму палярэднічае - пошуку ключавых крапак і падліку адпаведных дэскрыптараў.

У практычнай частцы я правёў уласнае даследванне прымяняльнасціых ці іншых дэтэктараў і дэскрыптараў, зрабіў высновы наконт ўплыву харектара выявы на атрыманы вынік.

Найбольшым унёскам у задачу рэканструкцыі стала реалізацыя прыкладання, якое здзяйсняе будоўлю мадэлі з набора выяваў. Програма знаходзіцца ў актыўнай распрацоўцы, праца над ёй працягваецца. Код знаходзіцца ў вольным доступе і можа быць знайдзены на GitHub.

Задача рэканструкцыі паверхні па дадзеных з БПЛА ўключае ў сябе мноства этапаў, кожны з якіх варты ўвагі і варты асобнага даследвання. У будучым я планую працягваць распрацоўку тэмы, сачыць за сучаснымі падыходамі да працэса рэканструкцыі, вылучаць асаблівасці задачы рэканструкцыі па дадзеных з БПЛА адносна агульнай задачай трохмернай рэканструкцыі аб'ектаў.

СПІС КРЫНІЦАЎ

- [1] Michal Irani and P. Anandan. About direct methods. In *Proceedings of the International Workshop on Vision Algorithms: Theory and Practice*, ICCV '99, pages 267–277, London, UK, UK, 2000. Springer-Verlag.
- [2] Edward Rosten and Tom Drummond. Machine learning for high-speed corner detection. In *Proceedings of the 9th European Conference on Computer Vision - Volume Part I*, ECCV'06, pages 430–443, Berlin, Heidelberg, 2006. Springer-Verlag.
- [3] David G. Lowe. Distinctive image features from scale-invariant keypoints. *Int. J. Comput. Vision*, 60(2):91–110, November 2004.
- [4] Herbert Bay, Andreas Ess, Tinne Tuytelaars, and Luc Van Gool. Speeded-up robust features (surf). *Comput. Vis. Image Underst.*, 110(3):346–359, June 2008.
- [5] Ethan Rublee, Vincent Rabaud, Kurt Konolige, and Gary Bradski. Orb: An efficient alternative to sift or surf. In *Proceedings of the 2011 International Conference on Computer Vision*, ICCV '11, pages 2564–2571, Washington, DC, USA, 2011. IEEE Computer Society.
- [6] Michael Calonder, Vincent Lepetit, Christoph Strecha, and Pascal Fua. Brief: Binary robust independent elementary features. In *Proceedings of the 11th European Conference on Computer Vision: Part IV*, ECCV'10, pages 778–792, Berlin, Heidelberg, 2010. Springer-Verlag.
- [7] Stefan Leutenegger, Margarita Chli, and Roland Y. Siegwart. Brisk: Binary robust invariant scalable keypoints. In *Proceedings of the 2011 International Conference on Computer Vision*, ICCV '11, pages 2548–2555, Washington, DC, USA, 2011. IEEE Computer Society.
- [8] Christian Forster, Matia Pizzoli, and Davide Scaramuzza. SVO: Fast semi-direct monocular visual odometry. In *IEEE International Conference on Robotics and Automation (ICRA)*, 2014.
- [9] Chris Sweeney. Theia multiview geometry library: Tutorial & reference. <http://theia-sfm.org>.