UNIVERSITATEA ”ALEXANDRU-IOAN CUZA” DIN IAS, I

**FACULTATEA DE INFORMATICA˘**



LUCRARE DE LICENT, A˘

**Object tracking**

propusa˘ de

**Mihai-Cătălin Bujor**

**Sesiunea:** iulie, 2019

### Coordonator s, tiint, ific

UNIVERSITATEA ”ALEXANDRU-IOAN CUZA” DIN IAS, I

**FACULTATEA DE INFORMATICA˘**

**Event Advertiser**

**Mihai-Cătălin Bujor**

**Sesiunea:** iulie, 2019

### Coordonator s, tiint, ific

Avizat, Iˆndruma˘tor lucrare de licent, a˘, Conf. Dr. Ionescu Ionel.

Data: ............................ Semna˘tura: ............................

#### Declarat, ie privind originalitatea cont, inutului lucra˘rii de licent, a˘

Subsemnatul **Mihai-Cătălin Bujor** domiciliat ˆın **Romaˆnia, jud. Vaslui, mun. Vaslui, strada Ștefan cel Mare bl. 173, et. 3, ap. 5**, na˘scut la data de **14 noiembrie 1997**, identificat prin CNP **1234567891234**, absolvent al Faculta˘t, ii de informatica˘, **Facultatea de informatica˘** specializarea **informatica˘**, promoția 2019, declar pe propria ra˘spundere cunoscaˆnd consecint, ele falsului ˆın declarat, ii ˆın sensul art. 326 din Noul Cod Penal s, i dispozit, iile Legii Educat, iei Nat, ionale nr. 1/2011 art. 143 al. 4 s, i 5 referitoare la pla- giat, ca˘ lucrarea de licent, a˘ cu titlul **Object tracking** elaborata˘ sub ˆındrumarea doamnei **Lect. Dr. Ignat Anca**, pe care urmeaza˘ sa˘ o sust, in ˆın fat, a comisiei este originala˘, ˆımi apart, ine s, i ˆımi asum cont, inutul sa˘u ˆın ˆıntregime.

De asemenea, declar ca˘ sunt de acord ca lucrarea mea de licent, a˘ sa˘ fie verificata˘

prin orice modalitate legala˘ pentru confirmarea originalita˘t, ii, consimt, ind inclusiv la

introducerea cont, inutului ei ˆıntr-o baza˘ de date ˆın acest scop.

Am luat la cunos, tint, a˘

despre faptul ca˘

este interzisa˘

comercializarea de lucra˘ri

s, tiint, ifice ˆın vederea facilita˘rii falsifica˘rii de ca˘tre cumpa˘ra˘tor a calita˘t, ii de autor al unei lucra˘ri de licent, a˘, de diploma˘ sau de disertat, ie s, i ˆın acest sens, declar pe proprie ra˘spundere ca˘ lucrarea de fat, a˘ nu a fost copiata˘ ci reprezinta˘ rodul cerceta˘rii pe care am ˆıntreprins-o.

Data: ............................ Semna˘tura: ............................

#### Declarat, ie de consimt, a˘maˆnt

Prin prezenta declar ca˘

sunt de acord ca lucrarea de licent, a˘

cu titlul **Object**

**tracking**, codul sursa˘ al programelor s, i celelalte cont, inuturi (gra- fice, multimedia, date de test, etc.) care ˆınsot, esc aceasta˘ lucrare sa˘ fie utilizate ˆın cadrul Faculta˘t, ii de informatica˘.

De asemenea, sunt de acord ca Facultatea de informatica˘ de la Universitatea

”Alexandru-Ioan Cuza” din Ias, i, sa˘

utilizeze, modifice, reproduca˘

s, i sa˘

distribuie ˆın

scopuri necomerciale programele-calculator, format executabil s, i sursa˘, realizate de mine ˆın cadrul prezentei lucra˘ri de licent, a˘.

Absolvent **Mihai-Cătălin Bujor**

Data: ............................ Semna˘tura: ............................

# Cuprins

[Motivat, ie](#_bookmark0) 2

[Introducere](#_bookmark1) 3

1. Colectarea Datelor de test 5

[1.1](#_bookmark3) Criteriile ce au stat la baza selecției . . . . . . . . . . . . . . . . . . . . . 6

1.2 Restricție privind datele de intrare . . . . . . . . . . . . . . . . . . . . . . 7

1. Metodele de urmărire folosite 8

[2.1 Boosting tracker](#_bookmark6) . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 8

[2.2 Mil tracker](#_bookmark7) . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 10

[2.3 Kcf tracker](#_bookmark8)  . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 12

2.4 Median-Flow tracker . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 14

2.5 Csrt tracker . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 16

1. Studii de caz 17

[3.1](#_bookmark11) Obstacole . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 17

[3.2](#_bookmark11) Dimensiune . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 19

[3.3](#_bookmark11) Schimbări de lumină . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 22

[3.5](#_bookmark11) Concluzii . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 23

1. Tehnologii folosite. 26

[4.1](#_bookmark11) Modulele aplicației . . . . . . . . . . . . . . . . . . . . . . . . . . . . 26

[4.2](#_bookmark11) Arhitectură . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 27

[Concluzii](#_bookmark12) 30

[Bibliografie](#_bookmark13) 31

# Motivat,ie

Într-o lume în care timpul reprezintă cea mai importantă resursă oamenii folosesc tehnologia pentru a automatiza tot mai multe procese din viața de zi cu zi fie că e vorba de muncă fizică ce este pusă pe seama roboților, activități din domeniul medical , securitatea sau chiar activitățile ce au drept scop relaxarea.

Odată cu avansul tehnologic producția autoturismelor a devenit un proces mult mai rapid astfel că numărul automobilelor a crescut simțitor în ultimii ani aceasta ducând la o supraaglomerare în special în zonele urbane. Astfel fie că e vorba de spații comerciale, aeroporturi sau chiar zonele din preajma locuințelor fiecăruia dintre noi căutarea unui loc de parcare devine o activitate risipitoare de timp și energie de care ne lovim din ce în mai des în ultima perioadă.

“Event-Advertiser” propune o alternativă de rezolvare a acestei probleme folosind tehnici din domeniul prelucrării de imagini. De-a lungul timpului au existat mai multe abordări pentru rezolvarea acestei probleme precum folosirea senzorilor de mișcare, dar această abordare vine cu dezavantajul dat de costuri suplimentare pentru resurse fizice mult mai mari . Pe de altă parte pentru a putea beneficia de funcționalitățile oferite de aplicație utilizatorul trebuie să dispună doar de o cameră de filmat alături de o conexiune la internet. Mai mult decât atât deși în stadiul actual aplicația este menită să servească câte un singur utilizator pe rând ea poate fi dezvoltată ulterior pentru folosirea sa în paralel astfel încât resursele hardware să fie partajate la rândul lor de mai mulți oameni ceea ce implică reducerea costurilor.

Principala motivație din spatele acestei lucrări este legată de interesul crescut pentru tehnicile din domeniul prelucrării de imagini care oferă o soluție pentru o gamă largă de probleme cu care ne confruntăm zilnic.

# Introducere

Urmărirea video reprezintă procesul de localizare a unui obiect aflat în mișcare pentru un interval timp folosind o cameră. Aceasta prezintă o varietate de utilități în viața de zi cu zi dintre care menționăm supravegherea, realitate augmentată sau controlul traficului. Scopul urmăririi video este de a realiza o asociere a obiectelor țintă pe parcursul cadrelor consecutive. Totuși această asociere poate fi uneori deosebit de dificilă atunci când videoclipul prezintă “zgomote” precum obstacolele, schimbările de luminozitate sau de viteză. Cu toate acestea dacă e să facem o comparație între urmărire și detecție, un alt proces ce face parte din domeniul prelucrării imaginilor constatăm că urmărirea este o metodă mai rapidă. Acest lucru este datorat faptului că ea presupune cunoașterea unor informații suplimentare despre obiectul țintă ce sunt extrase din cadrul precedent precum locația viteza sau direcția de deplasare. Un algoritm de urmărire va folosi astfel toate aceste informații pentru a realiza predicția în timp ce un algoritm de detecția începe mereu de la început fără nicio informație suplimentară. Aceste informații se pot pierde în situații în care obiectul devine complet acoperit de un altul pentru o perioadă mai îndelungată de timp sau dacă viteza sa de deplasare este prea mare pentru urmăritor Pentru a putea rezolva aceste probleme este necesară rularea unui algoritm de detectare la un anumit interval de timp. Precizia ridicată a acestui algoritm este datorată antrenării acestuia cu un set mare de date asemănătoare cu subiectul.

Un alt aspect important legat de urmărire este legat de faptul că aceasta păstrează identitatea spre deosebire de procesul de detecție care oferă la ieșire de cele mai multe ori o matrice de dreptunghiuri corespunzătoare obiectelor identificate fără a păstra însă o evidență a acestora.

Din moment ce am urmărit obiectul până într-un cadru anume putem afla și alte informații despre acesta nu doar cele legate direcție și respectiv viteza ci și despre modul cum arată acest obiect așadar putem construi un model de aspect ce are rolul de a limita regiunea de căutare. Mai mult decât atât putem considera acest model drept un clasificator menit să facă diferența între regiunea dreptunghiulară specifică obiectului și respectiv cea corespunzătoare fundalului. Acesta returnează un scor între 0 și 1 pentru a indica probabilitatea ca noul cadru să conțină sau nu obiectul. Scorul este 0 atunci când suntem absolut siguri de absența acestuia.

Aplicația “Event-Advertiser” își propune să folosească aceste tehnici de urmărire pentru a salva timpul irosit de un utilizator pentru găsirea unui loc de parcare .

# Capitolul 1

**Colectarea datelor de test**

Primul pas în procesul dezvoltării aplicației a fost reprezentat de colectarea datelor de test ce sunt reprezentate de fișiere în format video (.mp4, .avi etc) . Selecția acestora s-a făcut folosind setul de date de pe platforma “YouTube 8M” precum și date de test din proiecte cu tematici similare. Ulterior selecției datelor si măsurătorilor realizate am conceput un standard privind dimensiunea datelor de intrare pentru a putea oferi rezultate cât mai exacte pentru fiecare din metodele folosite, astfel fiecare video primit la intrare este redimensionat la următoarele valori: (lățime 1280p, lungime 680p).

## Criteriile ce au stat la baza selecției

În procesul de colectare a datelor de intrare am urmărit ca acestea să îndeplinească anumite criterii pentru a acoperi o gamă cât mai diversă de cazuri pentru test :

* să prezinte un subiect aflat în mișcare sau ce urmează a se pune în mișcare
* să prezinte zgomote ce sunt reprezentate de obstacole , schimbări de lumină sau fluctuații de viteză.
* să prezinte un subiect diversificat din punct de vedere al dimensiunii, culorii și poziției.
* filmulețul să fie realizat folosind o cameră stabilă.

## Restricție privind datele de intrare

Pentru a beneficia de rezultate cât mai bune metodele folosite de aplicație prezintă o restricție privind datele de intrare ce constă în faptul că filmulețul dat de utilizator trebuie realizat cu o cameră stabilă. Această condiție este necesară deoarece detecția subiectului respectiv a fundalului se realizează în fiecare cadru din datele de intrare alături de predicția privind următoarele posibile poziții ale subiectului (“bounding-boxuri”). „Bounding-boxurile” reprezintă coordonatele dreptunghiului ce înglobează o imagine în care se poziționează obiectul ce se dorește a fi urmărit într-un anumit cadru. Dacă pe parcursul filmării subiectul nu mai este prezent în cadru  “bounding-boxul” dispare de asemenea. Cazul în care filmarea este realizată cu o cameră aflată în mișcare (în care diferența dintre cadrele consecutive este mare) conduce la calcularea greșită a viitoarelor posibile poziții deoarece predicția se realizează folosind informațiile referitoare la poziția inițială a subiectului din secvența precedentă. Cu cât diferența dintre cadre este mai mare cu atât riscul de a pierde urma subiectului crește.

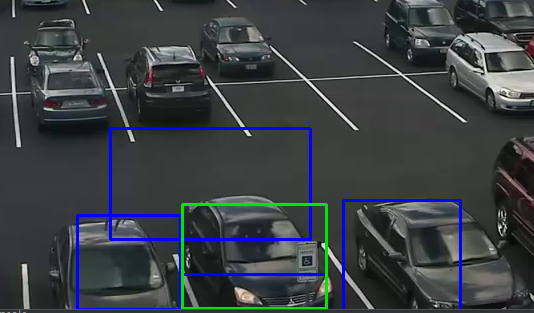


Figura 1.1

În figura de mai sus este prezentat modul în care se realizează predicția viitoarelor poziții (dreptunghiurile de culoare culoarea albastră) pentru un “bounding-box” oferit la intrare de către utilizator (dreptunghiul de culoare verde). După cum se poate observa metoda folosită estimează toate cele 4 modalități de deplasare posibile pentru subiect (înainte , înapoi, stânga ,dreapta) datorită faptului că nu se cunoaște poziția sa inițială și nici direcția de deplasare . Folosind această modalitate prevenim si cazurile în care subiectul (pentru cazul de mai sus) s-ar afla orientat în altă direcție. (de exemplu orizontal).

# Capitolul 2

**Metodele de urmărire folosite.**

Procesul inițial de dezvoltare a constat în utilizarea și efectuarea unor metrici asupra a 5 metode din biblioteca OpenCV :

* Boosting Tracker (OpenCV 3.0.0)
* Mil Tracker (OpenCV 3.0.0)
* CSRT Tracker (OpenCV 3.4.2)
* KCF Tracker (OpenCV 3.1.0)
* MedianFlow Tracker (OpenCV 3.0.0)

## Boosting Tracker

Este una dintre primele metode de urmărire dezvoltate ce are la bază o variantă “on-line” a algoritmul AdaBoost.[nr bibliografie] Diferența dintre versiunea “on-line” si cea “off-line” a algoritmului este dată de modul în care se obțin datele și de modul în care este construit clasificatorul puternic. Versiunea de antrenament “off-line” beneficiază de toate datele din avans în timp ce versiunea “on-line” folosește câte un eșantion pentru fiecare iterație. Construcția clasificatorului în versiunea de antrenament “off-line” se face adăugând un clasificator slab în fiecare rundă pe când în versiunea “on-line” clasificatorul este inițializat la început și actualizat după fiecare eșantion. Pentru a înțelege modul de funcționare a acestui algoritm este necesar să definim următorii termeni :

**Clasificator slab:** Acest tip de clasificator obține rezultate puțin mai bune decât selecția aleatoare, de exemplu pentru o problemă binară de decizie rata erorii trebuie să fie mai mică decât 50%). Ipoteza slabă obținută de un clasificator slab corespunde unei trăsături și se obține prin aplicarea unui algoritm de învățare.

**Selector:** Dată o mulțime de M clasificatori slabi ce au următoarele ipoteze:

Selectorul alege exact una din acestea , unde m este ales în funcție de criteriul de optimizare.

( , unde reprezintă eroarea estimată pentru fiecare clasificator slab).

**Clasificator puternic:** Dat un set de N clasificatori slabi, un clasificator puternic este calculat printr-o combinație liniară de selectori. Mai mult valoarea de încredere (o notăm cu „conf”) poate fi considerată drept o măsurătoare pentru acest tip de clasificator.

Ideea principala a acestui algoritm este introducerea selectorilor. Aceștia sunt inițializați în mod aleatoriu, iar fiecare din ei deține un grup separat de clase de clasificatori slabi. La întâmpinarea unui nou eșantion de formare , clasificatorii slabi ai fiecărui selector sunt actualizați. Această procedură de actualizare este cea care necesită cel mare timp de procesare. Considerând toate acestea problema de urmărire poate fi formulată ca o problemă binară de clasificare prin actualizarea continuă a clasificatorului actual al subiectului. Așa cum am menționat mai sus procesul de urmărire a subiectului face presupunerea că detecția sa a fost deja realizată. Regiunea inițială ( “bounding-boxul” ) dat la intrare de către utilizator este considerat imagine pozitivă pentru urmăritor. Totodată sunt stabilite și regiunile negative (de aceeași dimensiune) ca fiind regiunile învecinate cu acesta ce vor alcătui ulterior fundalul. Cu ajutorul acestora putem itera pentru a obține un prim model care este deja stabil. Procedura de antrenare selectează clasificatorul cu eroarea estimată minimală pentru a face parte din clasificatorul puternic. Pentru a putea estima eroarea clasificatorilor slabi se consideră 𝞴 denumit importanța unui eșantion ce este propagată prin setul de N selectori. În faza inițială se consideră 𝞴=1. Valoarea acestuia crește dacă eșantionul este greșit clasificat și invers altfel. În final clasificatorul puternic se obține sub forma unei combinații liniare formată din selectori.

În contextul urmăririi clasificatorul execută 4 pași :

1. Dată o poziție în runda i a subiectului acesta stabilește zona de căutare conform Figurii 2.1.
2. Pentru fiecare subzona ce aparține zonei de căutare atribuie o pondere conform procedurii de mai sus . (Inițial ponderile sunt egale pentru toate vecinătățile)
3. Determină vecinătatea cu ponderea cea mai bună.
4. Actualizează clasificatorul cu noua pondere.

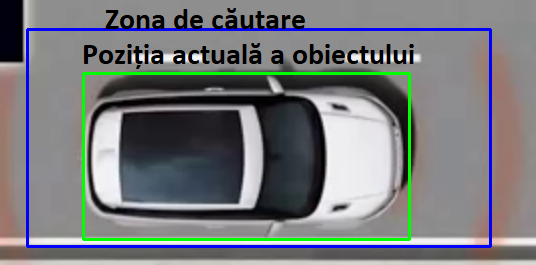


Figura 2.1

Datorită modalității de actualizare a urmăritorului prin care se urmărește identificare a celui mai slab clasificator, identificarea erorii pentru această metodă se face foarte greu de pildă în cazul în care subiectul prezintă fluctuații semnificative de viteză sau cazul în care subiectul este acoperit de către un alt obiect ce apare într-un anumit cadru.

Totuși pentru a putea măsura capacitatea metodei am conceput un alt mod de identificare al erorii și anume am considerat drept eroare cazul în care poziția rămâne constantă.

## Mil Tracker

Mil (“Multiple instance learning”) reprezintă o metodă de învățare supervizată din învățarea automată. Aceasta presupune găsirea unei funcții ce realizează o asociere între datele de intrare și cele de ieșire în baza unor alte perechi de tipul (date de intrare, date de ieșire) ce sunt considerate date de antrenament (de aici și numele de învățare). Scopul acestei proceduri este de a aproxima această funcție astfel încât aceasta să fie capabilă să clasifice noi date de intrare (date de test ) în mod corect.

Dacă în mod tradițional se utilizează un set de instanțe ce sunt clasificate individual MIL se folosește de o mulțime de astfel de instanțe clasificate împreună ce poartă numele de (“bags”). Cu alte cuvinte pentru a putea estima în mod tradițional este necesar un set de date de forma unde reprezintă o instanță (în cazul nostru un vector format din imaginile corespunzătoare vecinătăților) și eticheta instanței. În timp ce pentru MIL datele de antrenament au forma unde un „bag” } , iar reprezintă eticheta “bag-ului” definită astfel :

Cum sunt clasificate aceste mulțimi? În cazul cel mai simplu al clasificării binare o astfel de mulțime se clasifică negativ dacă toate instanțele sale sunt negative, respectiv pozitiv dacă aceasta conține cel putin o instanță clasificată pozitiv. Pentru a putea clasifica “bag-urile” ne propunem să antrenăm un clasificator de tip “boosting” pentru a maximiza funcția de verosimilitate a mulțimilor.

Având în vedere faptul că această funcție este definită pentru “bag-uri”, iar scopul nostru este de a obține un clasificator ce estimează , trebuie să exprimăm (probabilitatea ca un “bag” să fie clasificat pozitiv) în funcție de fiecare instanță,iar pentru asta recurgem la modelul următor :

Această ecuație are proprietatea că dacă una din instanțe are probabilitate mare atunci probabilitatea întregului “bag” este de asemenea mare.

În contextul urmăririi dificultatea constă în selectarea datelor ce reprezintă un exemplu pozitiv pentru urmăritor respectiv negativ. Cele mai multe metode printre care și „boosting-ul„(prezentată mai sus ) folosesc drept exemplu pozitiv locația curentă a subiectului în timp ce vecinătățile acestuia sunt clasificate drept instanțe negative. Dacă în schimb locația inițială nu este precisă modelul este actualizat în baza unor date parțial corecte. Ca urmare după un număr de iterații de acest tip există posibilitatea ca urmăritorul să piardă urma obiectului țintă.

Pe de altă parte o altă abordare constă în utilizarea mai multor exemple pozitive formate din cele mai apropiate vecinătăți ale subiectului. În cazul nostru aceste exemple pozitive sunt reprezentate de “bounding-boxurile” din jurul fiecărui obiect ce se dorește a fi urmărit urmând ca mai apoi aceste date să fie trimise către algoritmul de învățare pentru a se decide care dintre ele este cel mai corect (această procedură are loc pentru fiecare din mulțimile de instanțe după următorul pseudocod):

1. Algoritmul determină vecinătățile (“bounding-boxurile”) reprezentate sub forma de vectori folosind o raza de căutare notată cu reprezintă locația subiectului în secvența .
2. Clasifică vecinătatea reprezintă o variabilă binară ce indică prezența sau absența obiectului în imaginile corespunzătoare vecinătăților acestuia.
3. Actualizează locația urmăritorului folosind o strategie lacomă (“greedy”) dată de = astfel nu menținem o distribuție asupra locațiilor din raza a subiectului și oferim șansă egală   “tracker-ului “ la timpul
4. Formează două mulțimi } și } ce reprezintă o actualizare a modelului ce alcătuiește fundalul.
5. Actualizează modelul cu un “bag” pozitiv și negative fiecare conținând câte o singură imagine din mulțimea .

Utilizarea unei astfel de metode implică de asemenea și dezavantaje deoarece modelul poate deveni confuz după un număr de runde ceea ce reduce puterea de decizie a clasificatorului și totodata și timpul de execuție. Datorită faptului că această metodă încearcă să optimizeze mereu cel mai slab clasificator face ca identificarea posibilelor erori să se facă într-un mod mai dificil.

## Kcf Tracker

Kcf (Kernelized Correlation Filters) este o metodă ce utilizează tehnici similare cu cei

doi algoritmi prezentati mai sus plecând de la observația că eșantioanele pozitive folosite de MIL prezintă zone de suprapunere ce pot fi modelate sub forma unor matrici circulare.

Matricile circulare reprezintă un model de matrice în care fiecare diagonală descendentă de la stânga la dreapta este constantă și în care fiecare linie este rotită un element în dreapta față de linia anterioară.

Am notat cu diagonala matricii ce conține coeficienții transformării discrete Fourier al vectorului a, iar U este matricea unitară. Această modalitate de reprezentare ne permite să utilizăm operații Fourier în locul celor clasice pentru a putea optimiza timpul de execuție. De asemenea cu ajutorul acestor matrici putem recurge la un model de regresie “ridge regression” având drept scop găsirea unei funcții prin care dorim să minimizăm erorile pentru eșantioanele și regresiile lor

Am notat cu parametrul de regularizare ce controlează fenomenul de suprapunere. Aceasta apare ca o eroare de modelare atunci când o funcție este prea apropiată de un set limitat de puncte din setul de date. este numit minimizator și are următoarea formă unde reprezintă o matrice a căror linii sunt reprezentate de eșantioane, iar fiecare element este o țintă de regresie , iar este matricea identitate.

Modul de calcul pentru în forma de mai sus este mult prea costisitor având o complexitate exponențială ) , așadar pentru a optimiza soluția putem rescrie folosind domeniul Fourier drept , iar este conjugata lui .

În contextul „tracking-ului” funcția de regresie este antrenată cu toate shiftările eșantionului dat la intrare . Acest lucru se face deoarece dorim să obținem robustețe chiar si în cazul în care obiectul urmărit este supus evaluării. Pentru un vector de dimensiune m, matricea circulară este o matrice formată din toate shiftările lui a. Aceasta are proprietatea că poate reprezenta convoluția . Convoluția reprezintă o operație matematică peste două funcții prin care se produce o a treia funcție ce indică modul în care forma unei funcții este modificată de cealaltă.

Pentru a obține o mai bună discriminare este necesar să evaluăm cele de mai sus folosind caracteristici dimensionale superioare. În ecuația de regresie sunt necesare numai rezultatele multiplicării acestor vectori .Folosind formula de calcul de mai sus reducem timpul de execuție doar la a calcula transformarea discretă Fourier ce are o complexitatea . Așa cum este prezentat în Figura 3 modul de funcționare a KCF poate fi împărțit în cinci blocuri cu rolul de detecție, antrenare creare și actualizare a modelului.

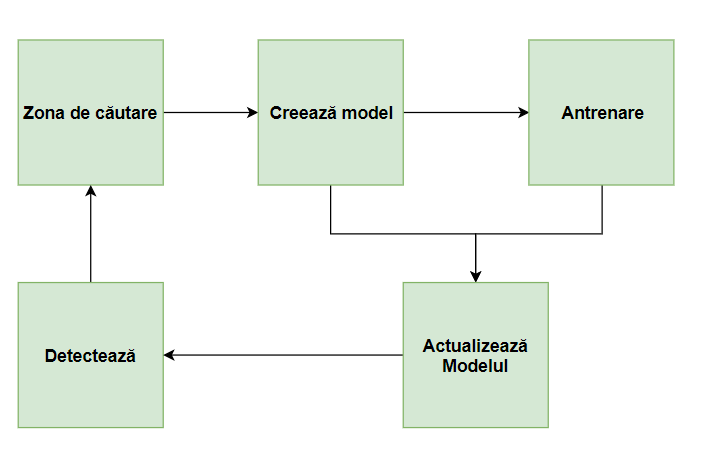


Figura 2.2 Modul de funcționare a metodei KCF

## MedianFlow Tracker

Această metodă de urmărire are la bază așa numita “forward-backward consistency” ce pleacă de la ipoteza conform căreia urmărirea corectă trebuie să fie independentă de fluxul de timp. Putem sintetiza modul de funcționare sub forma unui algoritm ce presupune următorii pași:

1. Urmăritorul produce o traiectorie pentru subiect : considerăm o secvență de imagini și locația subiectului în frame-ul t. Acest subiect este urmărit  înainte pe parcursul celor k secvențe ceea ce produce traiectoria definită prin am notat cu f modul de urmărire înainte “forward”.
2. În ultimul cadru din această traiectorie se produce o validare ce constă în urmărirea subiectului în sens invers (de la ultimul frame către primul). Scopul nostru este de a estima eroarea acestei traiectorii în secvența de imagini . Pentru acesta construim traiectoria de validare, punctul este urmărit înapoi până la primul frame: unde .
3. Următorul pas presupune compararea celor două traiectorii, iar dacă acestea diferă semnificativ predicția se consideră drept eronată. Eroarea deci se poate defini prin distanța dintre cele două traiectorii . Putem defini mai multe moduri de calcul pentru această distanță de exemplu cea Euclidiană definită prin :

Așa cum am menționat mai sus ipoteza FB (“forward-backward”) se bazează pe faptul că aceste puncte sunt independente. În realitate acest caz se întâmplă foarte rar, totuși putem observa că punctele formează blocuri ce se mișcă împreună. La rândul lor blocurile se pot identifica drept obiecte. Așadar urmăritorul primește ca date de intrare o pereche de imagini și un “bounding-box” (ce reprezintă de fapt un bloc) și produce la ieșire “bounding-boxul” .

Pentru a putea urmări blocul se folosește metoda Lucas-Kenade ce va genera fluxul mișcării între cele două perechi de imagini . Metoda Lucas-Kenade este o metodă de estimare a debitului optic. Aceasta presupune că debitul este în esență constant într-o vecinătate locală a pixelului examinat și rezolvă ecuațiile de bază ale fluxului optic pentru toți pixeli vecini folosind metoda celor mai mici pătrate. Un aspect foarte important este legat de faptul că această metodă pleacă de la presupunerea că diferențele de luminozitate între cadrele consecutive sunt foarte mici.

Calitatea predicției punctelor este ulterior estimată, urmând ca fiecare din puncte să primească o eroare. Următorul pas presupune eliminarea a jumătate din cele mai slabe puncte, iar cealaltă jumătate este folosită pentru prezicerea întregului obiect (“box”). Estimarea deplasării cutiei delimitate de punctele rămase se efectuează folosind valoarea mediană peste fiecare dimensiune spațială. Această dimensiune se calculează pentru fiecare pereche de puncte calculând distanța dintre punctul curent si cel anterior . Părțile obiectului care nu se găsesc în acest interval nu sunt luate în considerare în procesul de votare.

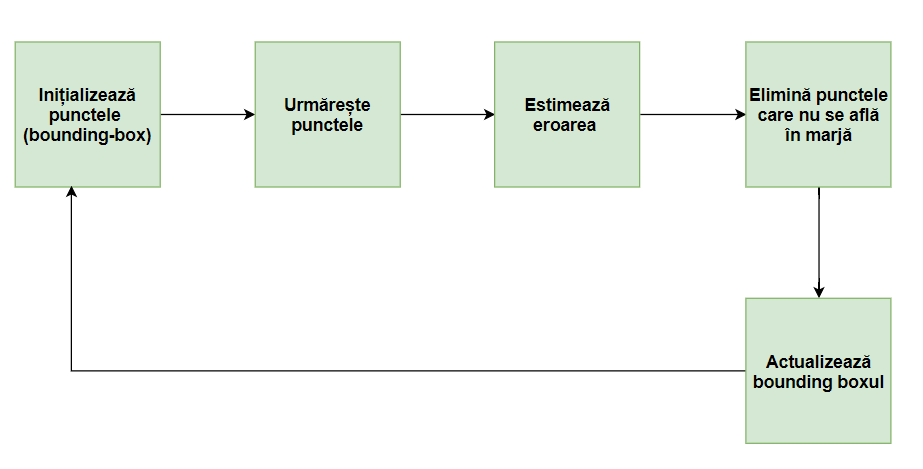


Figura 2.3 Modul de funcționare a metodei Median-Flow

MedianFlow este o metodă prin care prezența erorile este ușor raportată datorită pasului de verificare pe care aceasta îl execută .Totodată cazuri precum cele în care viteza subiectului este prea mare fac ca acest model de urmărire să eșueze.

## CSRT Tracker

CSRT este o metodă bazată pe filtre de corelare la fel ca si în cazul kcf diferența constă în faptul că acesta se folosește în plus de o hartă de fiabilitate spațială ce are ca scop ajustarea suportului pentru filtre . Prin folosirea acestei hărți se constată o reducere a impactului pe care îl poate avea fundalul asupra obiectului ce se dorește a fi urmărit. Acest lucru asigură mărirea și localizarea regiunii selectate și o mai bună urmărire a regiunilor și obiectelor ce nu au dimensiuni rectangulare. Metodele de corelare discriminative învață un filtru cu un răspuns predefinit pe imaginea de antrenament . Aceasta din urmă este obținut prin extinderea ușoară a regiunii din preajma subiectului pentru a include și eșantionarea de fundal. La baza acestei metode se află două caracteristici standard numite hog și nume de culori .

Histograma gradienților orientați (hog) este un descriptor al caracteristicilor utilizat în domeniul prelucrării de imagini pentru a detecta o anumită țintă. Aceasta numără practic toate aparițiile gradientului în porțiunile localizate ale unei imagini. Gradientul reprezintă o schimbare direcțională a intensității sau culorii dintr-o imagine.

Numele de culori sunt etichete lingvistice pe care oamenii le folosesc pentru a comunica culoarea. Calcularea numelui de culori învață o mapare de la valorile pixelilor la numele de culori.

Fiabilitatea canalului este o altă noutate pe care o introduce această metodă de urmărire . Estimarea fiabilității se face prin două modalități învățare respectiv verificare. Partea de învățare constă în folosirea proprietăților soluției constrânse a celor mai mici pătrate din proiectarea filtrului. Metoda celor mai mici pătrate este o metodă matematică de a obține o soluție a unui sistem de ecuații supradeterminat adică care are mai multe ecuații decât necunoscute. Soluția este obținută prin minimizarea sumei pătratelor abaterilor față de valorile ecuațiilor. Partea de verificare se reflectă în modul care un singur canal votează pentru o posibilă locație a obiectului țintă.

Partea de urmărire constă de asemenea în doi pași localizarea și respectiv actualizarea. În partea de localizare caracteristicile sunt extrase dintr-o zonă de căutare ce este centrată pe poziția anterioară a subiectului. Obiectul este localizat prin însumarea acestor caracteristici obținute din canalul de fiabilitate. Pasul de actualizare presupune că regiunea de căutare este centrată la locația țintă estimată la etapa precedentă.

# Capitolul 3

**Metrici realizate asupra metodelor folosite**

Pentru a putea determina o metodă cât mai bună ce acoperă un număr cât mai mare de cazuri am separat datele de test în categorii peste care am efectuat ulterior diverse măsurători. Acestea au vizat performanța ca timp de execuție cât și numărarea erorilor produse atât pentru metodele clasice oferite de biblioteca open-cv cât și îmbinări ale acestora. Pe axa orizontala a fiecărui grafic de mai jos se regăsesc numele metodelor pentru care s-au executat testele, iar pe cea verticala valorile ce au rezultat pentru aceasta în funcție de categorie astfel : albastru reprezintă timpul total de execuție, verde timpul epuizat până la detecția locului liber de parcare, iar roșu este reprezentativ pentru numărul erorilor apărute pe parcursul execuției În secțiunile din acest capitol dorim să prezentăm rezultatele obținute în urma acestor măsurători și situațiile pe care acestea le acoperă:

## 3.1 Date de test ce prezintă obstacole

Aplicația funcționează într-un context în care prezența obstacolelor este des întâlnită mai mult de atât acestea sunt de diverse tipuri și dimensiuni spre exemplu putem considera drept obstacol un om care apare în fața autovehiculului sau chiar o altă mașină. Este foarte important ca metoda de urmărire folosită să fie capabilă să urmărească subiectul în ciuda noului obiect apărut  în cadru pentru a oferi astfel disponibilitate permanentă și o experiență cât mai plăcută utilizatorului . În situația în care obiectul devine complet acoperit pentru un interval de timp mai ridicat informațiile ce alcătuiesc modelul de aspect sunt transmise către algoritmul ce execută detecția. Acesta iterează timp de câteva cadre consecutive în vederea recuperării obiectului. Dacă scorul returnat de acesta este 0 atunci se consideră că urmăritorul a pierdut urma subiectului.

# 

Figura 3.1

Graficul din Figura 3.1 ilustrează rezultatele testelor efectuate asupra celor 5 metode ce au rulat pe un set de date format din 7 videoclipuri în care dimensiunea obstacolului diferă, iar luminozitatea este constantă. Considerăm că un urmăritor a eșuat în momentul în care acesta pierde urma subiectului fără șansă de revenire. Se poate observa că pentru acest caz metoda “Multiple-Intance-Learning (MIL)” produce numărul cel mai redus de erori, dar și timpul de execuție respectiv detecție semnificativ mai ridicat. Aceasta se datorează faptului că MIL ia în considerare cele mai multe posibilități pentru a realiza predicția (prin formarea “bag-urilor”). Pe de altă parte metodele  “Kernelized Correlation Filter” și “Median Flow” sunt cele mai eficiente din punct de vedere al timpului de execuție, iar numărul de erori apropiat de celelalte metode.

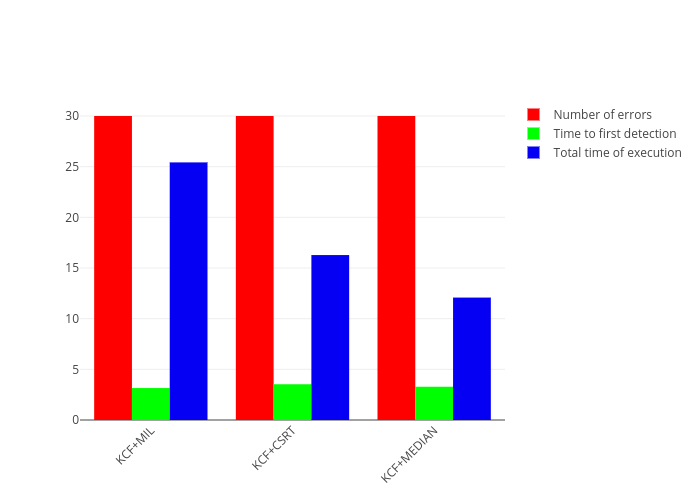


Figura 3.2

În urma observațiilor am decis că pot realiza îmbinări ale metodelor în funcție de cazul pentru care acestea oferă cele mai bune rezultate (Figura 3.2) având drept scop minimizarea erorilor și respectiv a timpului de execuție. Am folosit ca metodă de start kcf deoarece conform măsurătorilor de mai sus este cea mai rapidă metodă, iar acest lucru este foarte important la început pentru a putea detecta cu succes locul de parcare. Ulterior am executat teste pe același set de date îmbinând kcf mil, kcf csrt și kcf median. Lipsa din graficul de mai sus a metodei de tip boosting este datorată faptului că măsurarea erorilor pentru aceasta s-a făcut într-un mod diferit față de celelalte metode (deoarece aceasta are un raport al erorii foarte slab). Rezultatele testelor pe acest set de date arată faptul că numărul de erori este similar ceea ce înseamnă că metoda kcf a identificat cu succes subiectului și respectiv locul de parcare a reușit ulterior să-l urmărească până într-un anumit cadru.

Din punct de vedere al timpului de execuție îmbinarea cu median-flow oferă rezultatele cele mai bune pentru acest set de date, iar cea cu mil cele mai slabe.

## 3.2 Cazuri în care subiectul este diversificat din punct de vedere al dimensiunii.

Dimensiunea obiectului țintă este un alt factor foarte important pentru procesul de urmărire, aceasta poate varia de la un cadru la altul fie datorită faptului că acesta se îndepărtează de cameră fie pentru că este acoperit parțial sau complet de un alt obiect . De asemenea un subiect de dimensiuni reduse este mult mai dispus la ocluziune ceea ce ridică o problemă pentru majoritatea metodelor.

Pentru a putea acoperi ambele cazuri am realizat măsurători atât pentru obiecte de dimensiuni reduse (Figura 3.2) cât și pentru obiecte a căror dimensiune este mai mare față de celelalte din jur (Figura 3.3), ca și în cazul anterior luminozitatea este constantă pe parcursul filmării. Testele au fost realizate pe un set de date ce conține 5 respectiv 8 videoclipuri, iar rezultatele sunt ilustrare în graficele de mai jos:

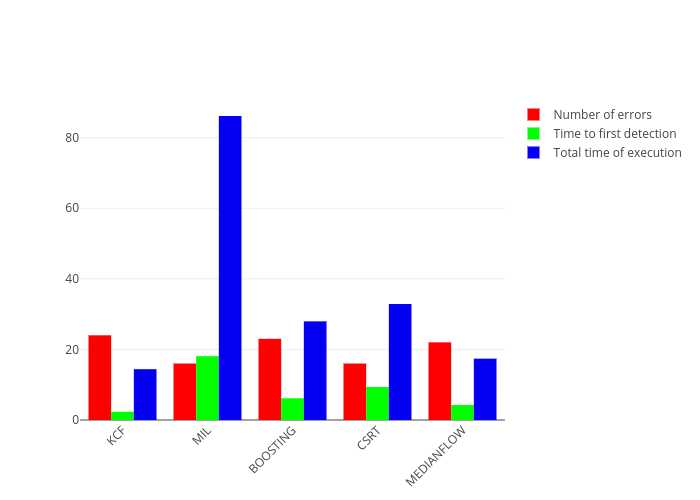


Figura 3.3 Graficul pentru obiecte de dimensiuni reduse

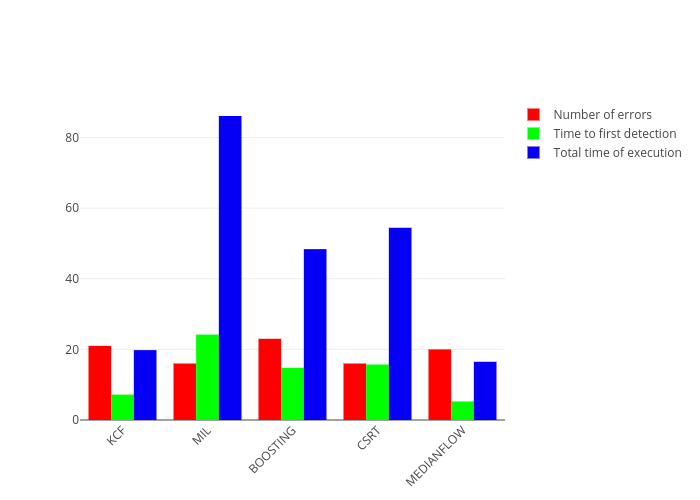


Figura 3.4 Graficul pentru obiecte de dimensiuni mari

Figura 3.3 ilustrează rezultatele obținute pentru obiecte a căror dimensiuni sunt reduse. Pentru acest caz KCF este metoda ce oferă cel mai mare număr de erori. Acest rezultat este datorat faptului că în situațiile în care subiectul este complet acoperit această metodă nu mai este capabilă să revină asupra obiectului țintă spre deosebire de cazul anterior în care subiectul era partial acoperit.

Ca timp total de executie pentru acest caz MedianFlow este metoda ce oferă cele mai bune rezultate. Fiind un obiect de dimensiuni reduse “bounding-box-ul” format de urmăritor are de asemenea dimensiuni reduse, deci este format si dintr-o mulțime cu număr limitat de puncte ceea ce eficentizează timpul de execuție al algoritmului.

În cazul obiectelor de dimensiuni mai mari MIL este metoda ce oferă cele mai puține erori, dar asemenea cazului anterior un timp de total execuție mult mai ridicat.

Din punct de vedere al timpului de execuție pentru acest studiu de caz metoda cu cele mai eficiente rezultate este Medianflow.

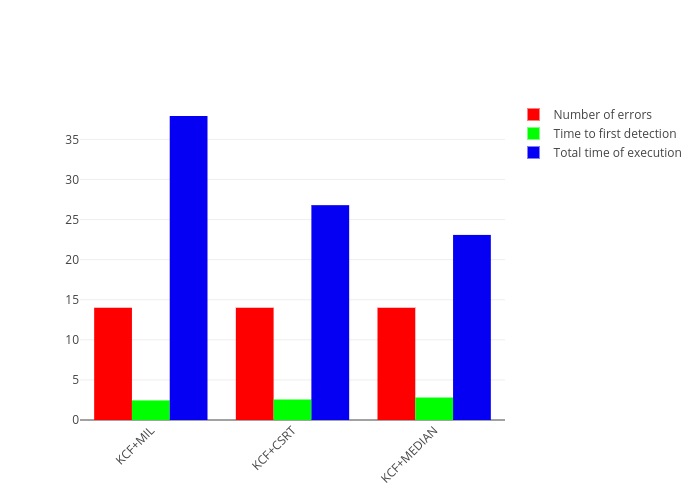


Figura 3.5

## 

Figură 3.6

Figurile 3.5 respectiv 3.6 ilustrează rezultatele obținute pentru metodele combinate pentru aceelași set de date. Ca si în cazul anterior am considerat drept metodă de start kcf datorită timpului de execuție mic pe care aceasta îl oferă. Dacă pentru obiectele de dimensiuni reduse Figura 3.5 numărul erorilor pentru metode este constant, iar timpul de execuție este descrescător de la stânga spre dreapta, în cazul subiecților de dimensiuni mai mari îmbinarea metodelor kcf și mil oferă cel mai mic număr de erori, dar totodata și cel mai mare timp de execuție. Metoda bazată pe boosting nu a fost inclusă în măsurători din aceleași considerente ca în cazurile anterioare.

## 3.3 Videoclipuri realizate  în condiții reduse de luminozitate

O altă categorie de zgomote ce apare des în datele de test este reprezentată de diferența de luminozitate. Pentru a oferi utilizatorului o experiență cât mai plăcută aplicația trebuie să poată să ofere disponibilitate și în condiții de utilizare speciale. Dacă subiectul este o mașină acest caz devine unul mai delicat datorită farurilor care produc un dezechilibru din acest punct de vedere.

Setul de date pentru acest caz este format din 4 filmulețe a unor parcări ce au fost realizate pe timp de noapte, în care camera este poziționată la diferite distanțe față de subiect și în diverse unghiuri, iar rezultatele sunt redate în graficul de mai jos:

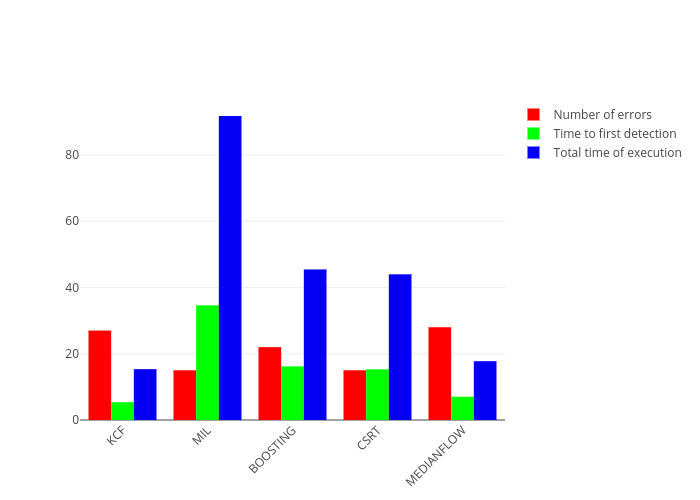


Figura 3.7

Pentru acest caz metoda care oferă cele mai multe erori este Medianflow deoarece aceasta pleacă de la ipoteza că luminozitatea rămâne constantă pe întreg parcursul videoclipului. Restul metodelor oferă valori destul de apropiate din punct de vedere al timpului de execuție. Totuși valoarea cea mai ridicată este obținută în cazul metodei MIL datorită modului în care aceasta calculează la fiecare cadru nou toate mulțimile de instanțe.

# 

Figura 3.8

Figura 3.8 ilustrează rezultatele obținute prin îmbinarea metodelor având ca metodă de start kcf. Se poate observa o scădere din punct de vedere a numărului de erori cât și a timpului de execuție, iar din punct de vedere al detecției rezultatele sunt constante ceea ce arată că metoda kcf a identificat cu succes locul de parcare respectiv subiectul și l-a urmărit până la un anumit cadru. Ca și în cazul anterior numărul maxim de erori este produs de către median-flow datorită ipotezei de la care pleacă acesta. Cea eficientă metodă pentru acest caz este îmbinarea dintre kcf si mil datorită timpului de execuție scăzut oferit de kcf și preciziei dată de mil .

## 3.4 Concluzii

Pentru a avea o imagine de ansamblu asupra tuturor situațiilor prezentate mai sus am realizat teste pentru toate cazurile anterioare atât pentru metodele clasice cât și pentru îmbinările lor. Trebuie menționat faptul că setul de date pentru care au fost executate testele nu este suficient de mare pentru a acoperi toate situațiile ce pot apărea în contextul aplicației. Totodată timpul de execuție poate varia în funcție de puterea de procesare a mașinii pe care au fost rulate testele.

# 

Figura 3.9 Metodele clasice

Rezultatele privind metodele clasice pentru acest set de date arată că tehnica bazată pe kcf este cea pentru care am obținut cele mai bune rezultate din prisma timpului de execuției și al detecției, iar ca număr de erori mil este metoda ce are precizia cea mai ridicată. Din acest motiv am ales să folosesc kcf drept metodă de start, urmând ca în caz de eroare aceasta să fie înlocuită de altă metodă de cu un timp de execuție mai slab, dar cu o precizie mai bună. Tranziția de la o metodă la alta se face în baza ultimilor pași executați de kcf (prin reținerea ultimelor coordonate ale “bounding-boxului”) și preluarea lor ulterioară de către noul urmăritor .

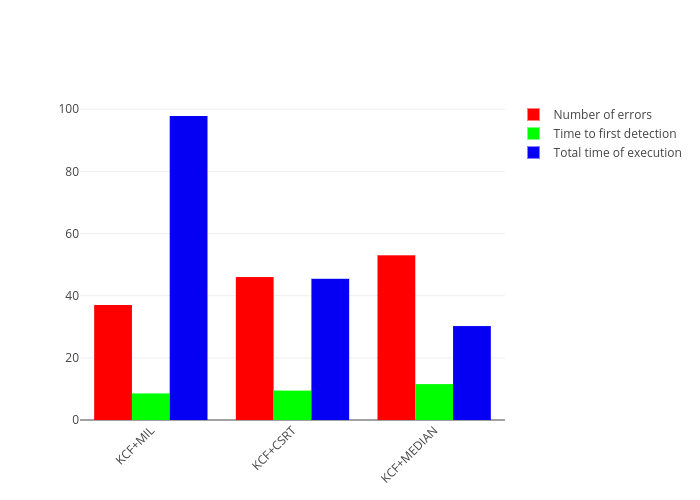


Figura 3.10

După cum se observă nu există o metodă ce minimizează atât numărul de erori cât și timpul de execuție așadar aplicația pune la dispoziție opțiunea de a alege metoda de rezervă în funcție de necesitățile utilizatorului. Prin urmare dacă urmărim să avem o precizie mai mare, dar suntem dispuși să sacrificăm timpul de execuție îmbinarea dintre metodele kcf și mil oferă cele mai bune rezultate. Datorită preciziei ridicate această metodă face față cazurilor în care obiectul execută mișcări bruște sau are o viteză de deplasare ridicată.

Pe de altă parte combinarea dintre metodele kcf și csrt respectiv median oferă un timp mult mai bun de execuție, dar totodată și o precizie mai redusă. Situațiile ce prezintă un avantaj prin această îmbinare sunt cele în care dorim să urmărim obiectul pe o distanță mare de timp la o viteză constantă.

# Capitolul 4

# Tehnologii folosite

# 

Acest capitol își propune să prezinte tehnologiile ce stau la baza aplicației cât și modul lor de funcționare. De asemenea dorim să prezentăm porțiuni cheie de cod semnificative pentru un anumit modul sau funcționalitate. Aplicația „Event-Advertiser” a fost dezvoltată folosind limbajul python deoarece acesta oferă suport pentru biblioteca open-cv și totodată și foarte multă flexibilitate.

Pentru a putea avea o separare a responsabilităților cât mai clară și pentru a facilita o dezvoltare ulterioară aplicația este dezvoltată folosind paradigma de programare orientată obiect. Astfel aplicația este împărțită pe mai multe module ce comunică între ele pentru a oferi funcționalitățile dorite de către cel ce le folosește. Aplicația dispune de 3 module mari ce au rolul de a urmărire a subiectului, notificarea utilizatorului în momentul în care un eveniment (în cazul de față un loc de parcare este eliberat ) are loc, respectiv o componentă ce oferă date cu privire la rezultatele metodelor de urmărire folosite. Ultima dintre aceste componente are un rol deosebit de important prin faptul că informațiile furnizate de ea ilustrează avantajele și dezavantajele fiecărei metode, iar asta poate duce ulterior la îmbunătățirea metodelor.

## 4.1 Modulele aplicației

Modulul de bază este cel ce realizează urmărirea respectiv detecția subiectului acesta folosește tehnicile descrise în capitolul 2 ce fac parte din librăria OpenCV. Aceasta este o librărie de procesare de imagini ce pune la dispoziție utilizatorului o gamă largă de algoritmi (peste 2500) pentru detecție, urmărire și chiar prelucrare de materiale video. OpenCv are o comunitate formată din 47 mii de utilizatori, iar numărul estimat de descărcări depășește 18 milioane. Biblioteca este utilizată extensiv de companii, grupuri de cercetare și organisme guvernamentale. Prezintă interfețe C++ , Python, Java , MATLAB și suportă Windows, Linux, Android și Mac OS. OpenCV-Python reprezintă extensia acestei librării peste limbajul Python ea fiind dezvoltată inițial in C++.

Python este un limbaj de programare cu scop general început de Guido van Rossum, care a devenit foarte popular într-un timp foarte scurt datorită simplității și lizibilității codului, acesta permite deci exprimarea unei idei în mai puține linii de cod, fără a reduce însă lizibilitatea, din acest motiv am ales să dezvolt aplicația folosind acest limbaj.

Modulul ce realizează trimiterea notificării are la bază o platformă de comunicații în cloud numită Twilio. Aceasta permite dezvoltatorilor să efectueze și să primească în mod programat apeluri telefonice, mesaje și alte funcții de comunicare utilizând API-urile de servicii web. Pentru a putea folosi această platformă este necesară crearea unui cont de utilizator, iar pentru efectuarea anumitor funcții de comunicare este necesară adăugarea un număr de credite pe cont. Twillio oferă suport pentru numeroase limbaje de programare precum: c#, java , pyhon , ruby , php, sau nodejs.

Graficele prezentate în capitolul 3 sunt realizate prin intermediul unul alt modul ce folosește instrumentele oferite de plotly. Aceasta platformă pune la dispoziție instrumente grafice de analiză și statistici precum și biblioteci de grafică științifică pentru limbaje precum python, r, matlab sau perl. Fiecare utilizator dispune de un panou central unde își poate stoca vizualiza și edita graficele. Gama pe care o oferă plotly este foarte diversificată suportând chiar si moduri personalizate de diagrame după necesitățile fiecarui utilizator. Plotly a fost construit folosind python și framework-ul django, iar pentru partea de front-end javascript alături de o bibliotecă de vizualizare d3.js. Fișierele tuturor utilizatorilor sunt găzduite de Amazon prin serviciul de stocare S3.

## 4.2 Arhitectura aplicației

Așa cum am menționat mai sus aplicația a fost dezvoltată folosind paradigma de programare orientată obiect pentru a oferi o decuplare și o delegare a responsabilităților cât mai benefică pentru o dezvoltarea ulterioară.

Prima interacțiune a utilizatorului cu aplicația este reprezentată de modulul prin care acesta introduce datele de intrare reprezentate de metoda de urmărire ce se dorește a fi folosită și respectiv modul de utilizare pe care acesta dorește să-l folosească. Aplicația poate opera în 2 moduri folosind un videoclip stocat local pe calculatorul utilizatorului sau folosind camera de filmat de pe dispozitivul acestuia. Când utilizatorul optează pentru folosirea camerei videoclipul nu este salvat local, dar datele cu privire la rezultatele obținute rămân stocate pe dispozitiv pentru generarea ulterioară a graficelor.

De asemenea pentru a putea beneficia de modulul de notificări utilizatorul poate introduce numărul său de telefon și datele contului pentru a permite accesul la platforma plotly. Pentru a oferi o experiență cât mai plăcută acest modul oferă de asemenea mesaje sugestive în cazul în care datele de intrare nu sunt introduse corespunzător. Ulterior preluării acestora un urmăritor ce folosește metoda bazată pe kcf este inițializat pentru a urmări subiectul. Mașina selectată de către utilizator “bounding-boxul” oferit la intrare este obiectul țintă al urmăritorului. În momentul în care acesta se pune în mișcare și eliberează un loc de parcare aplicația afișează un dreptunghi de culoare verde corespunzător locului liber. Detecția locului de parcare se face calculând distanța de la locul inițial la obiectul țintă în toate cele 4 direcții posibile. Odată cu afișarea locului disponibil este pornit un timer ce are rolul de a preveni cazul în care locul devine ocupat ulterior de o altă mașină. Când această perioada de timp ia sfârșit, iar locul rămâne în continuare liber utilizatorul este anunțat prin intermediul modulului de notificări ce trimite un mesaj către numărul de telefon primit la parametru. Pe de altă parte dacă pe parcursul urmăririi metoda kcf eșuează este creat un alt urmăritor corespunzător datelor de intrare oferite de utilizator. Dacă la rândul său acesta eșuează un mesaj de eroare este raportat.

Tot acest proces este supus măsurătorilor privind parametrii precum timpul de execuție și numărul erorilor, datele fiind stocate în fișiere ce sunt ulterior prelucrate și trimise către modulul de generare a graficelor (Grafic generator) care permite accesul la platforma plotly. Graficele generate sunt salvate în contul de utilizator de pe platformă și se pot descărca local sub diverse forme.

## 

# 

# Concluzii

Metodele de urmărire reprezintă o altă abordare pentru rezolvarea multor probleme cu care ne confruntăm în fiecare zi. Principalul avantaj al utilizării acestor metode este reprezentat de resursa hardware de care acestea au nevoie pentru a funcționa. Această ramură ce face parte din domeniul prelucrării de imagini se află încă la început, dar începe să fie utilizată în tot mai multe situații precum cele ce vizează securitatea, medicina sau eficientizarea asupra anumitor sisteme.

Aplicația “Event-Advertiser” a fost dezvoltată cu scopul de salva timpul pe care un utilizator îl irosește așteptând eliberarea unui loc de parcare. Modalitatea prin care obiectul țintă este urmărit este reprezentată inițial de kcf deoarece aceasta oferă cel mai bun raport dintre timpul de execuție și precizie, ulterior dacă acest tip de urmăritor pierde subiectul se recurge la o altă metodă selectată de utilizator.

Cu toate că aplicația oferă toate funcționalitățile pentru a rezolva problema enunțată mai sus ne putem gândi la direcții de viitor pentru aceasta atât pentru partea de urmărire cât și pentru partea de infrastructură.

Pentru urmărire o posibilă îmbunătățire este reprezentată de selectarea în mod automat a metodei de start ca urmare a executării unor algoritmi de detecție asupra videoclipului care să furnizeze informații legate de condițiile de lumină, tipul de obiect urmărit. Acest proces ar consuma mai multe resurse datorită numărului de calcule crescut ce trebuie executat de mașina de pe care rulează aplicația, dar ar oferi de altfel și o precizie mult mai bună. De asemenea această procedură se poate extinde și pentru selecția ce ține de metoda de rezervă folosită. Modulul de generarea a graficelor de care a dispune aplicația a fost dezvoltat tocmai pentru a oferi informații asupra metodelor actuale pe seturi mari de date cu scopul îmbunătățirii ulterioare.

Partea de infrastructură se poate extinde prin dezvoltarea aplicației pe platforme precum android sau ios ce oferă mai multă flexibilitate față de varianta actuală. De asemenea o altă direcție de îmbunătățire poate fi dezvoltarea unei variante care să fie capabilă să servească mai mulți utilizatori simultan ce pot selecta numărul dorit de obiecte țintă.

În concluzie această lucrare poate reprezenta un punct de pornire pentru dezvoltarea mai multor aplicații ce au la bază tehnici din domeniul prelucrării de imagini ce au drept scop salvarea cele mai importante resurse a omului, timpul.

# Bibliografie

* + - [1] Helmut Grabner, Michael Grabner and Horst Bischof. Real-time tracking via on-line boosting.
    - [2] J.F. Henriques, R. Caseiro, P.Martins and J. Batista. Exploiting the circulant structure of tracking-by-detection with kernels.
    - [3] Alan Lukezic, Tom'as Voj'ir, Luka Cehovin Zajc, Jir'i Matas, and Matej Kristan. Discriminative correlation filter tracker with channel and spatial reliability. *International Journal of Computer Vision*, 2018

* + - [4] Zdenek Kalal, Krystian Mikolajczyk, and Jiri Matas. Forward-backward error: Automatic detection of tracking failures. In *Pattern Recognition (ICPR), 2010 20th International Conference on*, pages 2756–2759. IEEE, 2010.
    - [5] Boris Babenko, Ming-Hsuan Yang, and Serge Belongie. Visual tracking with online multiple instance learning. In *Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on*, pages 983–990. IEEE, 2009.
    - [6] <http://cvgl.stanford.edu/papers/xiang_iccv15.pdf>
    - [7] [**http://www.robots.ox.ac.uk/~joao/publications/henriques\_eccv2012.pdf**](http://www.robots.ox.ac.uk/~joao/publications/henriques_eccv2012.pdf)
    - [8] <http://www.robots.ox.ac.uk/~joao/publications/henriques_tpami2015.pdf>
    - [9] <https://www.hindawi.com/journals/mpe/2018/6931020/>
    - [10] <https://www.twilio.com/docs/api>
    - [11] <https://www.pyimagesearch.com/2018/07/30/opencv-object-tracking/>
    - [12] <https://research.google.com/youtube8m/>
    - [13] <https://www.learnopencv.com/object-tracking-using-opencv-cpp-python/>
    - [14]