

Spis treści

1. Wizyjne śledzenie obiektów	3
1.1. Wprowadzenie	3
1.2. Ogólny model algorytmów śledzenia obiektów na obrazach.....	4
1.2.1. Reprezentacje obiektu na obrazie	5
1.2.2. Modele obiektu zainteresowania.....	6
1.2.3. Modele ruchu obiektu	7
1.2.4. Metody pomiaru podobieństwa oraz stwierdzenia dopasowania obiektu pomiędzy klatkami.	8
1.2.5. Metody aktualnienia modelu obiektu zainteresowania.....	10
2. Zagadnienia powiązane z wizyjnym śledzeniem obiektów.....	11
2.1. Ekstrakcja i deskrypcja cech obrazu.....	11
2.1.1. Punkty charakterystyczne	11
2.1.2. Krawędzie	15
2.1.3. Proste i krzywe matematyczne.....	17
2.2. Analiza ruchu.....	17
2.2.1. Ruch w sekwencji obrazów	17
2.2.2. Przepływ optyczny	20
3. Algorytmy śledzenia obiektów w robotyce mobilnej	25
3.1. Algorytm Lucasa-Kanade.....	25
3.1.1. Klasyczny algorytm Lucasa-Kanade	25
3.1.2. Algorytm Kanade-Lucas-Tomasi	27
3.1.3. Piramidowa wersja algorytmu Lucasa-Kanade.....	30
3.2. Algorytm <i>Mean Shift</i>	32
3.2.1. Algorytm <i>Mean Shift</i> w śledzeniu obiektów.....	32
3.2.2. Algorytm <i>CAMSHIFT</i>	36
3.3. Filtr Kalmana.....	37
3.3.1. Klasyczny filtr Kalmana	38
3.3.2. Rozszerzony filtr Kalmana.....	40
3.3.3. Filtr Kalmana w śledzeniu wizyjnym	42
4. Przegląd istniejących rozwiązań.....	45
4.1. Moduł sterowania kamerą <i>pan-tilt</i> dla pojazdu <i>UAV</i>	45

4.2.	Hybrydowy algorytm śledzenia i podążania za ludźmi dla robota kroczącego.....	46
4.3.	Algorytm wykrywania, śledzenia i podążania za obiektem na obrazie z kamery sferycznej dla robota mobilnego.....	46
4.4.	System wizyjnego śledzenia obiektów w czasie rzeczywistym do zastosowania w wizyjnym sterowaniu robota mobilnego.....	48
4.5.	System śledzenia obiektów z wykorzystaniem kamery <i>pan-tilt</i> zamontowanej na robocie mobilnym	49
4.6.	Podsumowanie.....	50

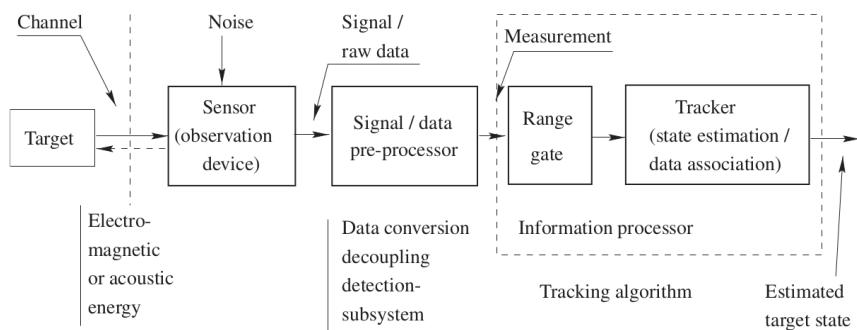
Spis rysunków

1.1	Ogólny model systemu śledzenia obiektów	3
1.2	Ogólny model działania algorytmów wizyjnego śledzenia obiektów	5
1.3	Reprezentacje obiektu zainteresowania na obrazie	6
1.4	Modele obiektu zainteresowania, stosowane w algorytmach śledzenia obiektów	7
1.5	Modele ruchu obiektu, stosowane w algorytmach wizyjnego śledzenia obiektów	8
1.6	Metody śledzenia obiektów na obrazach	9
2.1	Przykłady cech występujących na obrazach	12
2.2	Przykład ekstrakcji i deskrypcji punktów charakterystycznych metodą <i>SIFT</i>	15
2.3	Przykład wykrywania krawędzi z wykorzystaniem operatora Sobela	16
2.4	Przykład różnic pomiędzy kolejnymi obrazami sekwencji	18
2.5	Przykład wyniku operacji odejmowania dwóch kolejnych obrazów sekwencji	19
2.6	Warunki wystąpienia problemu apertury	19
2.7	Przykład występowania problemu korespondencji	20
2.8	Przykład wizualizacji przepływu optycznego	21
3.1	Przebieg algorytmu Lucasa-Kanade	27
3.2	Przykłady piramidy obrazów	31
3.3	Przykłady działania metody <i>Mean Shift</i>	33
4.1	Bezzałogowy helikopter - platforma testowa	45
4.2	Przebieg algorytmu wizyjnego śledzenia obiektu	47
4.3	Przykładowy obraz z kamery sferycznej	47
4.4	Porównanie działania algorytmu <i>MCAMSGPF</i> z innymi metodami	49
4.5	Schemat systemu sterowania głowicą <i>pan-tilt</i>	50

1. Wizyjne śledzenie obiektów

1.1. Wprowadzenie

Zadanie śledzenia obiektu jest zasadniczo problemem estymacji zmiennych stanu opisujących jego kinematykę (położenie, prędkość, przyspieszenie), przy czym pomiary, na podstawie których jest ona wykonywana cechują się dużą niepewnością [6]. Ogólny model systemu śledzenia przedstawiono na rysunku 1.1:



Rysunek 1.1: Ogólny model systemu śledzenia obiektów

Źródło: [6]

Składa się on ze śledzonego obiektu, czujnika dokonującego pomiaru pewnej charakteryzującej go wielkości, procesora sygnałowego wstępnie przetwarzającego wyniki pomiarów oraz procesora informacji, realizującego na podstawie wyników pomiarów estymacji stanu [6].

Szczególnym przypadkiem systemów śledzenia obiektów są systemy, które jako podstawowy czujnik wykorzystują kamerę wizyjną. W literaturze bywają one określone jako **moduły śledzenia wizyjnego** (ang. *video trackers* lub *vision trackers*). Ich zastosowanie w robotyce mobilnej stanowi przedmiot niniejszej pracy. Funkcją modułów śledzenia wizyjnego jest estymacja trajektorii obiektu zainteresowania na płaszczyźnie obrazu, co można rozumieć inaczej jako jego spójne etykietowaniu na kolejnych ramkach sekwencji video [28]. W przypadku robotów mobilnych, wyznaczona trajektoria najczęściej stanowi sygnał sprzężenia zwrotnego dla systemu sterowania, którego zadaniem jest utrzymanie obiektu zainteresowania w polu widzenia kamery - bez spełnienia tego wymagania wykonanie pomiaru stanowiącego podstawę estymacji nie jest możliwe. System sterowania może również realizować inne cele, jak np. zbliżanie się do obiektu zainteresowania lub utrzymywanie go w stałej odległości przy jednoczesnym unikaniu przeszkód, etc. Ogólnie, zastosowanie sygnału wizyjnego do kontroli ruchu robota nosi nazwę **serwosterowania wizyjnego** (ang. *visual servo control* lub *visual servoing*) [7].

Śledzenie obiektów w sekwencjach obrazów stanowi funkcję licznej grupy zróżnicowanych algorytmów przetwarzania obrazów, o wielu praktycznych zastosowaniach, wśród których można wymienić [28]:

- Wykrywanie obiektów na obrazie w oparciu o ich ruch
- Automatyczny monitoring (ang. *video surveillance*), tzn. wykrywanie i rejestrację podejrzanych zdarzeń
- Indeksację sekwencji video, tzn. ich oznaczanie i klasyfikację ich fragmentów
- Tworzenie interfejsów człowiek-komputer, poprzez rozpoznawanie gestów, okulografie, etc.
- Monitorowanie ruchu ulicznego, tzn. automatyczne gromadzenie danych statystycznych na temat jego natężenia
- Nawigację pojazdów, tzn. planowanie trasy oraz unikanie kolizji w oparciu o obraz z kamery video

Opcjonalnie, poza estymacją trajektorii, algorytm może dostarczać informacji o cechach samego obiektu, np. jego orientacji, rozmiarze, czy kształcie [28]. Tak określone zadanie w ogólnym ujęciu jest skomplikowane ze względu na szereg utrudniających je zjawisk, występujących w rzeczywistych sekwencjach video, takich jak [28]:

- Utrata części informacji o obiekcie, wynikająca z projekcji trójwymiarowej rzeczywistości na dwuwymiarową płaszczyznę obrazu
- Zakłóczenia występujące na obrazach rejestrowanych kamerą
- Skomplikowany charakter ruchu obiektów zainteresowania
- Brak spełniania warunku sztywności obiektów zainteresowania
- Częściowe i całkowite przysłonięcia obiektów zainteresowania
- Złożony kształt obiektów zainteresowania
- Zmiany oświetlenia sceny na obrazach
- Wymaganie czasu rzeczywistego obliczeń

Samo zagadnienie od dawna stanowi przedmiot badań, w związku z czym powiązane z nim pojęcia i problemy są dokładnie sklasyfikowane i zdefiniowane, zaś dotycząca go literatura jest bardzo obszerna. Tym niemniej ze względu na jego trudność opracowane dotychczas rozwiązania cechują się specjalizacją, tj. wykazują odporność na część z przedstawionych wyżej problemów, przy jednoczesnym jej braku na pozostałe [23].

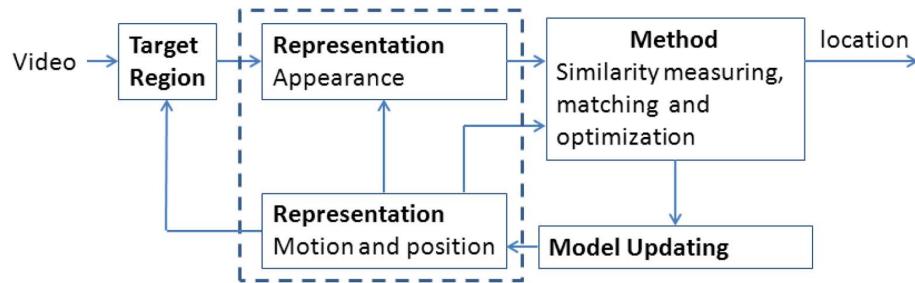
1.2. Ogólny model algorytmów śledzenia obiektów na obrazach

Choć algorytmy wizyjnego śledzenia obiektów są bardzo zróżnicowane, można dokonać ich dekompozycji na pięć podstawowych etapów [23]:

- Określenie obszaru występowania obiektu zainteresowania
- Wyznaczenie reprezentacji obiektu zainteresowania
- Wyznaczenie modelu ruchu obiektu zainteresowania

- Wykonanie pomiaru podobieństwa oraz określenie dopasowania obiektu pomiędzy kolejnymi klatkami sekwencji obrazów
- Uaktualnienie reprezentacji obiektu zainteresowania

Na rysunku 1.2 przedstawiono schemat kolejności realizacji tych etapów. Nie uwzględnia on fazy wykrywania obiektu, stanowiącej odrębne zagadnienie cyfrowego przetwarzania obrazów.



Rysunek 1.2: Ogólny model działania algorytmów wizyjnego śledzenia obiektów

Target region - Obszar zainteresowania (obszar obrazu zawierający obiekt zainteresowania)

Representation - appearance - Model obiektu zainteresowania określający jego cechy charakterystyczne

Representation - motion and position - Model ruchu obiektu zainteresowania

Method - similarity measuring, matching and optimization - metoda pomiaru podobieństwa oraz stwierdzenia dopasowania obiektu pomiędzy klatkami

Model updating - metoda aktualizowania modelu cech obiektu zainteresowania

Źródło: [23]

1.2.1. Reprezentacje obiektu na obrazie

Pierwszym krokiem po uzyskaniu informacji o wystąpieniu obiektu jest zdefiniowanie jego reprezentacji rozumianej tutaj jako wyodrębnione składowe elementy obrazu przynależące do tego obiektu.

W najprostszym przypadku obiekt przybliża się jego **bryłą brzegową**, o postaci prostokąta lub elipsy zawierającej wszystkie należące do niego piksele. Inaczej mówiąc reprezentację obiektu stanowi fragment obrazu, który go zawiera. Zaletą takiego rozwiązania jest minimalna liczba parametrów opisująca obiekt, wadą natomiast włączeniem do jego reprezentacji dużej liczby pikseli tła [23].

Pewnym ulepszeniem tej koncepcji, zwiększającym jej odporność, jest stosowanie **wielu brył brzegowych** jednocześnie. Takie rozwiązanie cechuje się również większą elastycznością w dalszych krokach algorytmu, w których realizowane jest poszukiwanie położenia w kolejnej klatce (uproszczone w przypadku pojedynczej bryły brzegowej do jednego punktu) [23].

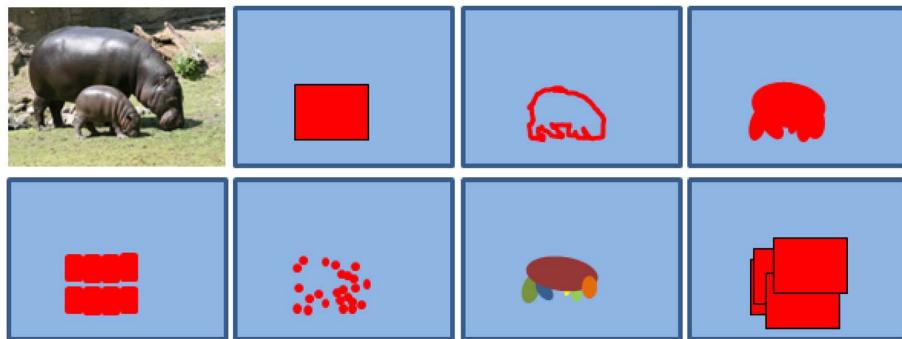
Inną możliwą, lecz rzadko stosowaną w tej rodzinie algorytmów reprezentacją jest **kontur** obiektu zainteresowania. Cechuje się ona małą rygorystycznością w stosunku do zmiany ogólnego wyglądu obiektu, jednak jej użycie jest ograniczone do przypadków, w których stosowanie deskryptorów kształtu zapewnia wysoką dokładność [23].

Pozornie podobny sposób reprezentacji polega na określeniu **ciąglego obszaru obrazu** zawierającego piksele należące do obiektu zainteresowania. Należy zwrócić uwagę, że taka reprezentacja obiektu nie jest równoznaczna jego sylwetce (ang. *silhouette*), rozumianej jako sumę zbiorów pikseli konturu obiektu oraz zbioru pikseli zawartych wewnętrz wzorcowanej przez niego bryły. Różnica ta jest wynikiem sposobu wyznaczania tych dwóch re-

prezentacji. Przedstawienie obiektu w postaci ciągłego obszaru obrazu pozwala na jego precyzyjne oddzielenie od innych obiektów. Wymaga natomiast dużej odporności od metody jego wyznaczenia, w celu uniknięcia wniesienia błędnej informacji o samym obiekcie [23].

Nieco bardziej złożona reprezentacja obiektu oparta jest o **skrawki obrazu**. Są to małe obszary obrazu zawierające fragmenty obiektu zainteresowania, którego wygląd może zmieniać się niezależnie w ich obrębie. Odmianą tego modelu jest reprezentacja w postaci punktów charakterystycznych (ang. *salient points* lub *interest points*), w których występuje pewna cecha charakterystyczna (ang. *feature*) obrazu. Istnieją różne koncepcje definicji zbioru oraz relacji pomiędzy skrawkami obrazu [23]. Cechą charakterystyczną takiej reprezentacji jest brak konieczności wyraźnego rozróżnienia pomiędzy tłem a obiektem zainteresowania. Skutkuje ona zwiększoną odpornością na przysłonięcia oraz zmianę kształtu obiektu zainteresowania. Z drugiej strony, w przypadku małych lub sztywnych obiektów reprezentacja skrawkami obrazu nie jest bardziej wydajna od reprezentacji opierających się o zbiory pojedynczych pikseli (jak np. w postaci bryły brzegowej) [23].

Na rysunku 1.3 przedstawiono przykładowy obiekt zainteresowania oraz wizualizacje odpowiadających mu reprezentacji według omówionych koncepcji.



Rysunek 1.3: Reprezentacje obiektu zainteresowania na obrazie

Kolejno, od lewego górnego rogu: obraz wejściowy, bryła brzegowa (ang. *bounding box*), kontur obiektu, obszar obiektu (ang. *blob*), skrawki obrazu (ang. *image patches*), rozproszony zbiór punktów zainteresowania, części obiektu, wiele brył brzegowych

Źródło: [23]

1.2.2. Modele obiektu zainteresowania

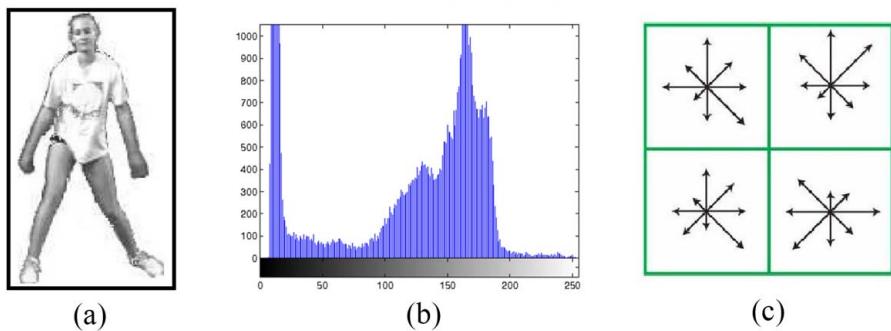
Model obiektu, rozumiany jako opis jego cech charakterystycznych, tworzony jest w obrębie zdefiniowanego wcześniej obszaru zainteresowania (1.2.1). Podstawowym wymaganiem, które musi spełniać jest zdolność do zachowania określonej właściwości pomiędzy kolejnymi klatkami, bez której wykonanie zadania śledzenia obiektu nie jest możliwe [23].

Istnieją trzy podstawowe reprezentacje cech charakterystycznych obiektu stosowane w algorytmach śledzenia: dwuwymiarowa tablica wartości pikseli, histogram oraz wektor cech [23].

Najbardziej oczywistą i bezpośrednią jest reprezentacja w postaci **dwuwympiarowej tablicy wartości pikseli obiektu** (rysunek 1.4a). Najczęściej w tablicy przechowywane są wartości określające jasności poszczególnych pikseli [23], pomimo idącego za tym założenia dotyczącego jej stałych wartości, które rzadko jest prawdziwe. Tym niemniej taka reprezentacja nie powoduje utraty informacji o obiekcie wewnątrz obszaru zainteresowania.

Reprezentacja w postaci **histogramu** (rysunek 1.4b) wyznaczana jest najczęściej w oparciu kolor pikseli w obszarze zainteresowania, występuje również w wersji przechowującej jedynie ich jasności [23]. Przy reprezentacji w tej postaci następuje całkowita utrata informacji o właściwościach przestrzennych obiektu, przechowywana jest jedynie informacja o wartościach poszczególnych pikseli, co skutkuje największą elastycznością w odniesieniu do zmiany kształtu obiektu [23].

Reprezentacja w postaci **wektora cech** (rysunek 1.4c) polega na wykryciu oraz deskrypcji cech charakterystycznych obiektu (ang. *features*). W jej przypadku również nie przechowuje się informacji o kształcie geometrycznym obiektu [23]. Cechy wraz z ich deskryptorami odwzorowują obiekt w oparciu o jego najbardziej bogate w informację właściwości, pozwalając na ograniczenie rozmiaru jego reprezentacji [23].



Rysunek 1.4: Modele obiektu zainteresowania, stosowane w algorytmach śledzenia obiektów

a) Tablica dwuwymiarowa b) Histogram c) Wektor cech

Źródło: [23]

1.2.3. Modele ruchu obiektu

Przyjęcie założeń dotyczących ruchu obiektu (w tym sensie, stworzenie jego uproszczonego modelu) ma za zadanie zawężenie okna, w którym następuje próba stwierdzenia wystąpienia obiektu na kolejnej klatce sekwencji.

Najprostszy model nie narzuca żadnych ograniczeń poza założeniem, że obiekt po wykonaniu przemieszczenia pomiędzy kolejnymi klatkami znajduje się blisko pozycji początkowej [23]. Jego implikacją jest stwierdzenie możliwości lokalizacji kolejnego położenia obiektu poprzez przeszukanie małego okna wokół poprzedniego [23], skąd nazwa **przeszukiwanie jednorodne**. Dzięki braku założeń dotyczących charakteru samego ruchu, rozwiązanie to cechuje duża ogólna odporność, nie sprawdza się ono jednak, jeśli ruch obiektu jest szybki [23].

Rozwinięciem koncepcji opisanej wyżej jest **gaussowski model ruchu**, w którym wprowadza prawdopodobieństwo wystąpienia kolejnego położenia obiektu wokół położenia początkowego zmienia się zgodnie z rozkładem normalnym. W tej reprezentacji dopasowaniu obiektu na kolejnej klatce przypisywany jest dodatkowo współczynnik wagowy (w oparciu o prawdopodobieństwo), co pozwala na zwiększenie przestrzeni poszukiwania [23].

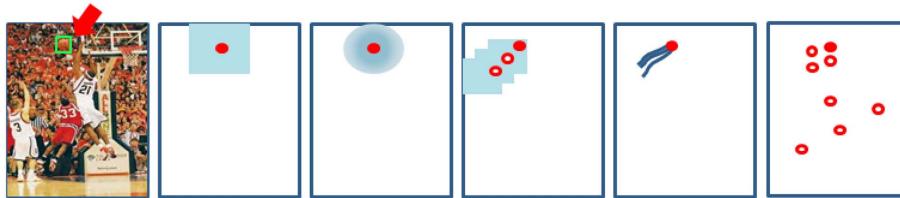
Innym możliwym modelem jest **predykcja ruchu** obiektu z wykorzystaniem jego liniowego modelu oraz filtra Kalmana [23]. Taką reprezentację można dodatkowo ulepszyć, wykonując predykcję w oparciu o przepływ optyczny [23].

W bezpośredniej predykcji ruchu nie przyjmuje się założeń odnośnie modelu ruchu, wykorzystuje się natomiast metody optymalizacji do znalezienia najlepiej dopasowanego nowego położenia w oparciu o położenia

występujące wcześniej [23]. Taki model jest jednak mało użyteczny, ponieważ wymaga, aby przemieszczenia obiektu były małe w porównaniu do innych zmian występujących na obrazie [23].

Ostatnia z metod reprezentacji polega na równoległym **śledzeniu i detekcji** obiektu z zastosowaniem niezależnych algorytmów. Po takim podejściu można oczekiwać dużej odporności na przypadkowe ruchu zarówno obiektu jak i kamery [23], jednak cechuje się ono największą komplikacją.

Na rysunku 1.5 zestawiono i porównano przedstawione wyżej sposoby modelowania ruchu.



Rysunek 1.5: Modele ruchu obiektu, stosowane w algorytmach wizyjnego śledzenia obiektów

Od lewej: obiekt zainteresowania, przeszukiwanie jednorodne, gaussowski model ruchu, predykcja ruchu, model wprost, śledzenie i detekcja

Źródło: [23]

1.2.4. Metody pomiaru podobieństwa oraz stwierdzenia dopasowania obiektu pomiędzy klatkami

Metoda pomiaru podobieństwa obiektu oraz stwierdzenia jego dopasowania pomiędzy klatkami to kluczowy element algorytmu śledzenia, które w tym właśnie etapie jest zasadniczo realizowane.

Pierwsza i najprostsza jej realizacja opiera się na wykonaniu **dopasowania** pomiędzy znanym modelem obiektu zainteresowania na początkowej klatce, do kandydatów, których modele tworzone są dla jego możliwych położen na kolejnej klatce, zgodnie z przyjętym modelem ruchu. Jednemu znanemu obiektowi zainteresowania jest więc przypisywany zbiór jego możliwych kolejnych instancji (rysunek 1.6a), spośród których wybierana jest najlepiej dopasowana. Inaczej mówiąc, śledzenie w tym przypadku jest zdaniem optymalizacji. Na podstawie podejścia do jego rozwiązania można wyróżnić dwie kategorie - dopasowanie bezpośrednie, polegające na poszukiwaniu najlepszego kandydata wzduż gradientu pewnej funkcji określającej dopasowanie, oraz dopasowanie probabilistyczne, polegające na znalezieniu maksimum funkcji gęstości prawdopodobieństwa, opisującej prawdopodobieństwo wystąpienia obiekt zainteresowania w danych punkcie obrazu. Dopasowanie bezpośrednie wykazuje dużą odporność przy wielu różnych modelach ruchu obiektu, przy zachowaniu warunku małej przestrzeni przeszukiwania. Z drugiej strony opiera się one o silne założenia dotyczące opisu modelu obiektu, w skutek jest wrażliwe na lokalne zmiany intensywności pikseli (występujące w rzeczywistych sekwencjach obrazów, w wyniku np. zmiany kąta padania światła) oraz jakość reprezentacji obiektu na obrazie [23]. Dopasowanie probabilistyczne jest efektywne w przypadku możliwości wystąpienia przysłonięcia obiektu oraz niejednoznaczności, pojawiających się np. przy wielu podobnych obiektach [23].

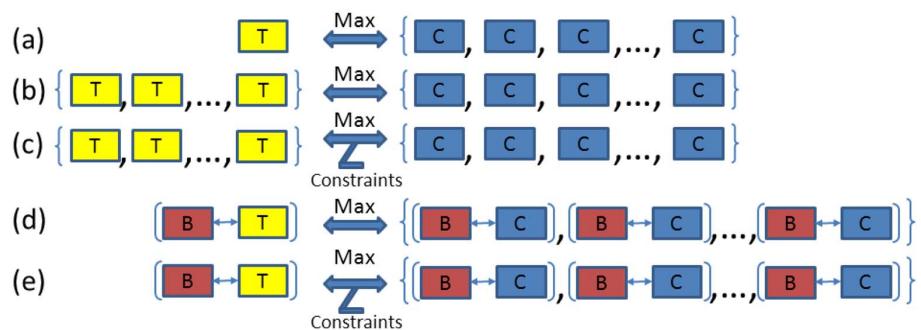
Dopasowanie z rozszerzonym opisem cech rozszerza koncepcję dopasowania o opis modelu lub stan metody w przeszłości (dla ciągu przednich klatek) [23]. Przy takim podejściu wieloelementowemu zbiorowi znanych wystąpień obiektu zainteresowania w ciągu poprzednich klatek przyporządkowywany jest wieloelementowy zbiór kandydatów nowej instancji tego obiektu na klatce bieżącej (rysunek 1.6b). Metoda przechowuje długotrwałe trajektorie stanu obiektu zainteresowania, w związku z czym jest użyteczne w zadaniach śledzenia długoterminowego.

wego oraz w przypadku możliwości wystąpienia przysłonięć [23]. Dopasowywanie jest wykonywane pomiędzy poszczególnymi elementami obydwu zbiorów, co skutkuje zwiększeniem złożoności obliczeniowej.

W celu jej ograniczenia wprowadzono **dopasowywanie z więzami**, przy którym podobieństwo pomiędzy zbiorami wystąpień i kandydatów jest rozpatrywane jedynie dla elementów spełniających zdefiniowane kryteria, lub inaczej, więzy (rysunek 1.6c). Mogą dotyczyć one, np. zmiany koloru lub kształtu obiektu pomiędzy kolejnymi klatkami. Metody tego typu mogą dobrze sprawdzać się w warunkach dużej zmienności obserwowanej sceny, nadają się również do zastosowania w algorytmach śledzących predefiniowanych wzorców [23].

Alternatywą dla metod opierających się o dopasowanie jest **klasyfikacja rozróżniająca**, określana inaczej jako „śledzenie przez wykrywanie” [23]. Opiera się ona o założenie, że samo odróżnienie pojedynczego obiektu zainteresowania od tła jest wystarczające do jego śledzenia [23]. Klasyfikacja rozróżniająca polega więc na określeniu przynależności wszystkich pikseli obrazu do tła bądź obiektu zainteresowania, oraz uaktualnianiu informacji o tej przynależności na kolejnych klatkach sekwencji video. W tej metodzie występuje więc przyporządkowanie pomiędzy oznaczonym obiektem zainteresowania i jego tłem poprzedniej klatce oraz zbiorem par kandydat-tło na kolejnej (rysunek 1.6d). Klasyfikator może przetwarzać zestaw wielu różnych cech jednocześnie, co pozwala na osiągnięcie dużej odporności w przypadku niskiej specjalizacji algorytmu śledzącego oraz wysokiej niestabilności obserwowanej sceny [23]. U podstaw metody leży jednak domniemanie unikalności cech śledzonego obiektu w skali obrazu, co powoduje jej błędne działanie w przypadku jego wielokrotnego wystąpienia [23]. Jest ona także wrażliwa na błędne etykietowanie pikseli, prowadzące do zakłócenia pracy klasyfikatora oraz przeklamania śledzonej trajektorii obiektu zainteresowania (ang. *drift*).

Klasyfikacja rozróżniająca z więzami jest rozwinięciem poprzedniej rodziny metod, wśród których główny nacisk kładzie się na poprawną klasyfikację i etykietowanie pikseli, a nie na poszukiwanie najlepszej (najbardziej dopasowanej lub prawdopodobnej) pozycji śledzonego obiektu [23]. Polega ono na syntezie informacji o przynależności zbioru pikseli do obiektu z poszukiwaniem w jego otoczeniu obszaru najlepiej dopasowanego do stworzonego uprzednio modelu, w postaci np. wektora cech (rysunek 1.6e). Wadą tej metody jest wysoka złożoność obliczeniowa, wynikająca ze stopnia jej komplikacji.



Rysunek 1.6: Metody śledzenia obiektów na obrazach

- a) Dopasowanie b) Dopasowanie z rozszerzonym opisem cech c) Dopasowanie z więzami d) Klasyfikacja rozróżniająca e) Klasyfikacja rozróżniająca z więzami. Oznaczenia bloków: T - wystąpienie obiektu zainteresowania, C - potencjalne kolejne wystąpienie obiektu zainteresowania, B - lokalne wystąpienia tła.

Źródło: [23]

1.2.5. Metody uaktualnienia modelu obiektu zainteresowania

Ostatni krok algorytmu śledzenia obiektu polega na uaktualnieniu stosowanego modelu na podstawie informacji zawartych w nowej klatce sekwencji video. W niektórych przypadkach nie jest on konieczny - poza uproszczeniem całego algorytmu, zaletą takiego rozwiązania jest brak wnoszenia błędnej informacji o obiekcie, która może być skutkiem niedoskonałego modelu [23].

Najprostsza realizacja uaktualnienia polega na **zastąpieniu** dotychczasowego opisu modelu opisem wykonanym dla ostatnio zaobserwowanego wystąpienia obiektu. Może się to odbywać w sposób częściowy, poprzez zastępowanie kolejnych fragmentów opisu nowymi, bądź przez okresowe, całościowe jego zastąpienie [23].

Predykcja postaci modelu przy użyciu filtra Kalmana (3.3) jest koncepcyjnie podobna, pozwalając jednocześnie na ograniczenie wpływu zakłóceń, zwłaszcza jeśli przebieg trajektorii obiektu jest łagodny. Gwałtowne jej zmiany skutkują jednak wniesieniem dodatkowego błędu [23].

W przypadku reprezentacji obiektu w postaci skrawków obrazu, uaktualnianie modelu polega na odpowiednim ich wstawianiu, usuwaniu, zmienieniu i przesuwaniu. Jest to zadanie złożone, dodatkowo rozwiązywane w oparciu o ograniczony zasób informacji [23].

Dla algorytmów wykorzystujących rozszerzony opis cech obiektów stosowana jest przyrostowa analiza głównych składowych [23]. Pozwala ona na przechowanie informacji o stanie modelu w przeszłości.

2. Zagadnienia związane z wizyjnym śledzeniem obiektów

Rekursywny filtr Bayesa

Ukryty łańcuch Markowa

SURF

2.1. Ekstrakcja i deskrypcja cech obrazu

Uaktualnić według [19]

Pojęcie **cechy obrazu** (ang. *image feature*) jest z natury nieprecyzyjne i ogólnie może oznaczać dowolną informację związaną z obrazem, która jest przydatna w ramach wykonywania określonego algorytmu. Funkcjonuje ono w literaturze również w nieco konkretniejszym ujęciu i według monografii [24] obiekty określane cechami obrazu należą do jednej z trzech podstawowych kategorii:

1. Punkty charakterystyczne (ang. *keypoint features, interest points* lub *corners*) i regiony charakterystyczne (ang. *blobs*)
2. Kontury obiektów na obrazie (ang. *edges*)
3. Proste oraz krzywe występujące na obrazie

Porównanie przykładowych cech obrazów należących do poszczególnych kategorii przedstawione zostało na rysunku 2.1. Ekstrakcja i deskrypcja cech jest jedną z podstawowych metod analizy obrazów i znajduje zastosowanie m.in. w algorytmach rozpoznawania obiektów, śledzenia obiektów, mozaikowania czy dopasowywania obrazów w stereoskopii. Celem wykonania takiej operacji jest wydobycie z dużej ilości informacji zawartej na obrazie jej istotnej części.

2.1.1. Punkty charakterystyczne

Przenieść opis SIFT do 3

Punkt charakterystyczny można rozumieć jako punkt na obrazie wyróżniający się spośród otoczenia, np. narożnik występujący w konturze obiektu widocznego na obrazie. Analogicznie, **region zainteresowania** posiada cechę charakterystyczną odróżniającą go od tła, np. ma jednolity kolor. W obu przypadkach stwierdzenie „charakterystyczności” odbywa się poprzez porównanie z otoczeniem. Porównanie to polega zwykle (w dużym uproszczeniu) na znalezieniu maksimów lokalnych pewnej funkcji ocenяcej „charakterystyczność” dla każdego punktu obrazu oraz jego lokalnego otoczenia. Po wykryciu zbioru cech wykonuje się ich **deskrypcje** (ang. *feature*



Rysunek 2.1: Przykłady cech występujących na obrazach

Od lewego górnego rogu: Punkty charakterystyczne (tutaj z wizualizacją ich przykładowych deskryptorów), regiony charakterystyczne, kontury obiektów, linie proste występujące na obrazie

Źródło: [24]

description), polegającą na wyznaczeniu ich formalnego opisu (zwykle w postaci wektora liczb) cechującego się zwiążnością, powtarzalnością oraz invariantnością [24], tzn. odpornością na przekształcenia afiniczne, zmiany oświetlenia, szum, etc. Odbywa się to w oparciu o piksele otoczenia punktu charakterystycznego lub piksele samego regionu charakterystycznego. Granica różniąca punkt charakterystyczny i region charakterystyczny jest więc bardzo niewyraźna, tym niemniej występuje w literaturze. Analizę z wykorzystaniem cech tego typu kończy ich **dopasowanie** (ang. *feature matching*) lub **śledzenie** (ang. *feature tracking*), np. pomiędzy dwoma kolejnymi klatkami sekwencji video lub obrazami jednej sceny zarejestrowanymi przy użyciu dwóch kamer. Model w postaci zbioru lokalnych cech, którymi są punkty charakterystyczne, nie wymaga żadnych założeń dotyczących wyglądu czy charakterystyki reprezentowanego obiektu jak całości, jak to ma miejsce np. w przypadku reprezentacji konturnem. Czyni to punkty charakterystyczne odpowiednią metodą opisu obiektu w przypadku występowania zmian oświetlenia, przysłonięć i innych zakłóceń występujących powszechnie w rzeczywistych, użytkowych sekwencjach video [26].

Ze względu na dużą użyteczność, algorytmy ekstrakcji i deskrypcji punktów charakterystycznych stanowiły w ostatniej dekadzie przedmiot intensywnych badań [26]. W literaturze zaproponowano wiele różnych podejść do tego zagadnienia, jednak jako najważniejszy przykład, dobrze obrazujący poszczególne etapy jego realizacji wymienia się algorytm *SIFT* (*Scale-Invariant Feature Transform*) [26] [24]. Wykryte przy jego pomocy punkty charakterystyczne cechują się invariantnością na zmiany skali i rotacji obrazu, a także częściowo na zmiany oświetlenia oraz zmiany widoku wynikającej z przesunięcia kamery rejestrującej scenę. Dodatkowo wykazują one dużą specyficzność (tzn. cechy odpowiadające poszczególnym fragmentom obrazu znacznie się między sobą różnią), zaś sam algorytm jest efektywny obliczeniowo [18]. Opis działania metody *SIFT* zostanie poniżej przytoczony w

całości.

Pierwszym krokiem algorytmu jest wyznaczenie reprezentacji obrazu w **przestrzeni skali** (ang. *scale space*), w której informacja przechowywana w jego pikselach zależy nie tylko od ich współrzędnych x, y ale również od skali s . Pozwala to na wykrycie punktów charakterystycznych, które pozostają stabilne przy zmianie tej skali [26]. Reprezentacja obrazu w przestrzeni skali wyznaczana jest zgodnie ze wzorem:

$$L(x, y, \sigma) = G(x, y, \sigma) * I(x, y) \quad (2.1)$$

$$G(x, y, \sigma) = \frac{1}{\sqrt{2\pi}\sigma^2} e^{-(x^2+y^2)/2\sigma^2} \quad (2.2)$$

gdzie:

- x, y – współrzędne piksela na obrazie
- $I(x, y)$ – wartość piksela o współrzędnych (x, y) na obrazie I
- $L(x, y, \sigma)$ – reprezentacja obrazu w przestrzeni skali w punkcie (x, y)
- σ – rozmiar maski dla filtru Gaussa, definiujący skalę s
- $G(x, y, \sigma)$ – wartość maski Gaussa o rozmiarze σ dla piksela o współrzędnych (x, y)
- * – operator konwolucji

Punkty charakterystyczne znajdują się w lokalizacjach wystąpień lokalnych minimów i maksimów reprezentacji obrazu w przestrzeni skali L . W celu ich wyznaczenia obliczana jest różnica obrazów w sąsiednich skalach $D(x, y, \sigma)$ [18]:

$$D(x, y, \sigma) = (G(x, y, k\sigma) - G(x, y, \sigma)) * I(x, y) = L(x, y, k\sigma) - L(x, y, \sigma) \quad (2.3)$$

gdzie:

- k – stały współczynnik (zwykle $k \in [1, 1; 1, 4]$)
- $D(x, y, \sigma)$ – Różnica filtrów Gaussa, w skrócie *DoG* (z ang. *Difference of Gaussian*)

Przeszukiwanie przestrzeni skali odbywa się poprzez wielokrotne wyznaczenie $D(x, y, \sigma)$, któremu towarzyszy zwiększenie σ o stałą wartość. Lokalne ekstrema wyznaczane są poprzez porównywanie wartości $D(x, y, \sigma)$ dla współrzędnych (x, y, σ) z punktami sąsiadującymi z punktem (x, y) w skali określonej przez σ oraz dwoma regionami o wymiarach 3×3 , współśrodkowymi z (x, y) , o sąsiednich skalach [18].

W ten sposób otrzymywany jest zbiór punktów charakterystycznych o określonych współrzędnych (x, y) oraz skali σ . Pozycja określona tymi współrzędnymi jest jednak niedokładna - punkt charakterystyczny może znajdować się pomiędzy pikselami (które stanowią próbki pomiarowe), dodatkowo jeden piksel w niższej skali odpowiada wielu pikselom w wyższej. Położenie ekstremów \hat{x} jest wyznaczane na podstawie interpolacji funkcją kwadratową wyznaczonej poprzez rozwinięcie w szereg Taylora funkcji $G(x, y, \sigma)$ [18]:

$$\hat{x} = -\frac{\partial^2 D^{-1}}{\partial \hat{x}^2} \frac{\partial D}{\partial \hat{x}} \quad (2.4)$$

gdzie:

- \hat{x} – wektor współrzędnych w przestrzeni skali

Dodatkowo zbiór ekstremów jest filtrowany - odrzucane są niestabilne punkty charakterystyczne o niskim kontraście [18].

$$D(\hat{\mathbf{x}}) = D + \frac{1}{2} \frac{\partial D^T}{\partial \mathbf{x}} \hat{\mathbf{x}} \quad (2.5)$$

Ekstrema o wartości bezwzględnej $|D(\hat{\mathbf{x}})|$, obliczonej na podstawie równania 2.5, mniejszej od wartości progowej 0.03 są usuwane ze zbioru punktów charakterystycznych [18].

Operacja *DoG* silnie reaguje na krawędzie obecne na obrazie, co jest zjawiskiem niepożądany, ponieważ ma charakter kierunkowy i skutkuje zdefiniowaniem niestabilnych punktów charakterystycznych [18]. W celu ich eliminacji wprowadza się drugi etap filtracji, polegający na wyznaczeniu hesjanu H (równanie 2.6) i porównaniu jego śladu i wyznacznika z wartością progową (równanie 2.7).

$$\mathbf{H} = \begin{bmatrix} D_{xx} & D_{xy} \\ D_{yx} & D_{yy} \end{bmatrix} \quad (2.6)$$

$$\frac{\text{Tr}(\mathbf{H})^2}{\text{Det}(\mathbf{H})} < \frac{(r+1)^2}{r} \quad (2.7)$$

gdzie:

r – wartość progowa

Osiągnięcie inwariantności na obrót obrazu uzyskuje się poprzez przypisanie punktom charakterystycznym orientacji [18]. Na podstawie skali σ punktu wyznacza się w jego środku maskę Gaussa o wariancji równej 1,5 jego skali, w obrębie której wyznacza się moduł (równanie 2.8) oraz orientację (równanie 2.9) gradientu dla każdego piksela (x, y) .

$$m(x, y) = \sqrt{(I_s(x+1, y) - I_s(x-1, y))^2 + (I_s(x, y+1) - I_s(x, y-1))^2} \quad (2.8)$$

$$\theta(x, y) = \arctan\left(\frac{I_s(x, y+1) - I_s(x, y-1)}{I_s(x+1, y) - I_s(x-1, y)}\right) \quad (2.9)$$

Obliczone orientacje są następnie kumulowane w histogramie z wagami określonymi poprzez odpowiadające im moduły oraz wartości okna Gaussa. Histogram składa się z 36 przedziałów odpowiadających 360 deg orientacji. Przedziały o największej liczebności odpowiadają dominującym orientacjom [18], przy czym punktowi przypisywana jest orientacja odpowiadająca najliczniejszemu przedziałowi oraz wszystkie inne orientacje, dla których liczebności wynoszą przynajmniej 80% maksymalnej (punkt może mieć wiele orientacji jednocześnie).

Na podstawie lokalnych orientacji wyznaczany jest również deskryptor punktu charakterystycznego. Na region o wymiarach 16×16 pikseli o środku w punkcie charakterystycznym nakładana jest maska Gaussa o wariancji równej połowie jego wielkości. Następnie region ten jest dzielony na tablicę 4×4 podregionów, w obrębie których następuje akumulacja orientacji w histogramach o 8 przedziałach (analogicznie jak w wyznaczaniu orientacji samego punktu, z wagami odpowiadającymi modułowi oraz wartości okna Gaussa) [18]. W ten sposób powstaje wektor zawierający $4 \times 4 \times 8 = 128$ elementów, opisujących punkt charakterystyczny. Końcowym etapem algorytmu jest wykonanie normalizacji, progowania ograniczającego wartość elementów do 0.2 (próg wyznaczony eksperymentalnie) oraz renormalizacji. W ten sposób uzyskiwana jest inwariantność na zmiany iluminacji [18]. Deskryptory pozwalają na wykonanie dopasowania pomiędzy punktami charakterystycznymi (np. wykrytymi na dwóch różnych ujęciach, przedstawiających ten sam obiekt), np. przy pomocy algorytmu k-najbliższych sąsiadów (ang. *k nearest neighbours*) z zastosowaniem metryki euklidesowej. Przykładowy wynik ekstrakcji i deskrypcji punktów charakterystycznych z wykorzystaniem algorytmu *SIFT* przedstawiono na rysunku 2.2.



Rysunek 2.2: Przykład ekstrakcji i deskrypcji punktów charakterystycznych metodą *SIFT*

Punkty charakterystyczne zlokalizowane są w początkach białych strzałek, wizualizujących ich dominujące orientacje (kierunek) oraz skale (długość). Na przykładowym obrazie wykryto 289 punktów charakterystycznych.

Źródło: [26]

2.1.2. Krawędzie

Pomimo swoich zalet, punkty charakterystyczne nie zawsze stanowią optymalny sposób reprezentacji cech obiektu. Istnieje liczna grupa zagadnień, w których istotna jest informacja o obiekcie jako o całości, w szczególności o jego właściwościach geometrycznych. Przykładowo, są one kluczowe w zadaniu automatycznego rozpoznawania znaków drogowych lub wykrywania produktów na przenośniku taśmowym w oparciu o szablony zdefiniowane w postaci ich modeli CAD [26]. W takich przypadkach analizę obrazu przeprowadza się w oparciu o wykryte na nim **krawędzie**.

Krawędzie można zdefiniować jako granice pomiędzy regionami obrazu o różnym kolorze, intensywności bądź teksturze [24]. Wykrywanie tych granic jest jednak zadaniem trudnym i stanowi osobną, złożoną dziedzinę cyfrowego przetwarzania obrazów noszącą nazwę segmentacji [24]. Alternatywnie, na potrzeby zadania ekstrakcji cech, krawędzie można zdefiniować w ujęciu „lokalnym” - jako wystąpienia nagłych zmian intensywności pikseli, czyli formalnie jako punkty, w których moduł lokalnego gradientu funkcji intensywności (równanie 2.10) przyjmuje duże wartości [24].

$$\mathbf{J}(\mathbf{x}) = \nabla I(\mathbf{x}) = \left(\frac{\partial I}{\partial x}, \frac{\partial I}{\partial y} \right)(\mathbf{x}) \quad (2.10)$$

gdzie:

$\mathbf{J}(\mathbf{x})$ – wartość gradientu funkcji intensywności dla współrzędnych \mathbf{x}

\mathbf{x} – wektor współrzędnych piksela (x, y)

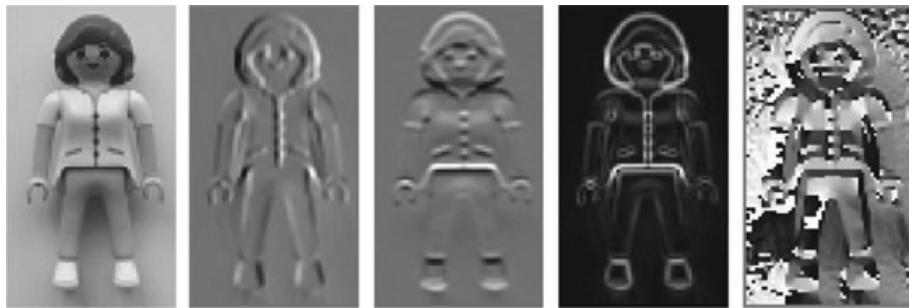
W praktyce obliczanie gradientu obrazu polega na wykonaniu jego splotu z odpowiednimi maskami, mającymi charakter kierunkowy. W wyniku takiej operacji otrzymuje się przybliżone pochodne kierunkowe obrazu, na podstawie których można następnie wyznaczyć moduł i orientację gradientu. Operacja wyznaczenia pochodnej obrazu wzmacnia wpływ składowych o wysokiej częstotliwości, w tym zakłóceń [24]. Rozmiar maski oraz waga jej elementów wpływają na zdolność filtru do uśredniania obrazu, a tym samym zmniejszania wpływu tych zakłóceń. Jednocześnie od tych parametrów zależy ogólne zniekształcenie obrazu (utrata informacji) i złożoność obliczeniowa całej operacji. W literaturze istnieje wiele propozycji masek o optymalnych właściwościach, jako przykład można podać maskę Sobela o rozmiarze 3×3 [26]. Jej postać dla kierunku poziomego $k_{S,x}$ oraz pionowego $k_{S,y}$ przedstawiono na rysunku 2.11.

$$\mathbf{k}_{S,x} = \frac{1}{4} \begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{bmatrix} \quad \mathbf{k}_{S,y} = \frac{1}{4} \begin{bmatrix} 1 & 2 & 1 \\ 0 & 0 & 0 \\ -1 & -2 & -1 \end{bmatrix} \quad (2.11)$$

Po obliczeniu gradientów kierunkowych I_x i I_y moduł I_G i orientację I_θ gradientu można wyliczyć z równania:

$$I_G = |I_x| + |I_y| \quad I_\theta = \arctan \frac{I_y}{I_x} \quad (2.12)$$

Przykład działania operatora Sobela przedstawiono na rysunku 2.3.



Rysunek 2.3: Przykład wykrywania krawędzi z wykorzystaniem operatora Sobela

Od lewej: obraz wejściowy w skali szarości; gradient poziomy (jasny kolor - wartości dodatnie, ciemny kolor - wartości ujemne); gradient pionowy (jasny kolor - wartości dodatnie, ciemny kolor - wartości ujemne); moduł gradientu; orientacja gradientu

Źródło: [26]

Innym sposobem usuwania zakłóceń wykonanie wstępnej filtracji dolnoprzepustowej, przy czym stosowany filtr powinien być kołowo-symetryczny [24] (powinien wygładzać obraz w każdym kierunku). Najczęściej wykorzystywany jest filtr Gaussa, posiadający tą właściwość. Ze względu na liniowość operacji różniczkowania i splotu, gradient filtrowanego obrazu można wyliczyć z równania 2.13 [24].

$$\mathbf{J}_\sigma(\mathbf{x}) = \nabla(G_\sigma(\mathbf{x}) * I(\mathbf{x})) = (\nabla G_\sigma)(\mathbf{x}) * I(\mathbf{x}) \quad (2.13)$$

gdzie:

G_σ – maska filtru Gaussa o wariancji σ

Gradient przefiltrowanego obrazu można więc wyznaczyć poprzez wykonanie jednej operacji konwolucji pomiędzy nim a poziomymi i pionowymi pochodnymi cząstkowymi maski Gaussa [24].

Częstym wymaganiem stawianym algorytmom wykrywania krawędzi jest ograniczenie zbioru wynikowego do krawędzi izolowanych, tj. zbioru pojedynczych pikseli zlokalizowanych wzdłuż krawędzi [24]. W celu wyodrębniania tych pikseli należy znaleźć w całym zbiorze pikseli krawędzi elementy o maksymalnym module gradientu w kierunku prostopadłym do samej krawędzi [24]. Realizacja tego zadania sprowadza się do wyznaczenia pochodnej kierunkowej obliczonego wcześniej gradientu wzdłuż jego kierunku (zgodnie z równaniem 2.14) oraz odnalezienia punktów zmiany jej znaku (ang. *zero crossing*) [24].

$$S_\sigma(\mathbf{x}) = \nabla \mathbf{J}_\sigma(\mathbf{x}) = \nabla^2 G_\sigma(\mathbf{x}) * I(\mathbf{x}) \quad (2.14)$$

gdzie:

∇^2 – laplasjan

$S_\sigma(\mathbf{x})$ – wartość funkcji znaku dla współrzędnych \mathbf{x}

Maska po wykonaniu splotu $\nabla^2 G_\sigma(\mathbf{x})$ nazywana jest laplasjanem filtru Gaussa (ang. *Laplacian of Gaussian*, w skrócie *LoG*). Ze względu na złożoność obliczeniową operacja *LoG* jest często aproksymowana przez przytoczoną wcześniej operacją *DoG* [24]. Wykrywanie zmiany znaku odbywa się poprzez porównanie wartości funkcji znaku S_σ dla dwóch sąsiednich pikseli. W przypadku jej wystąpienia, na podstawie współrzędnych tych pikseli określone jest położenie punktu należącego do krawędzi (może mieć on charakter sub-pikselowy), oraz, ewentualnie interpolowana jest wartość jego gradientu [24]. Punkt sklasyfikowany jako należący do krawędzi bywa w literaturze określany jako *edgel* [24] (połączenie słów *edge* i *element*).

Przedstawione wyżej metody wykrywania krawędzi operują jedynie na obrazach w skali szarości. Trywialnym sposobem rozszerzenia dziedziny ich działania na obrazy kolorowe jest osobne wykonanie dla każdej składowej koloru. Może to jednak prowadzić do wystąpienia niepożądanych zjawisk, takich jak np. wzajemne znoszenie się gradientów różnych składowych przy ich sumowaniu [24]. Istnieją bardziej zaawansowane algorytmy rozwiązania tego problemu, nie zostaną jednak omówione w niniejszej pracy.

Kolejnym zagadnieniem związanym z wykrywaniem krawędzi jest łączenie pikseli krawędzi izolowanych w ciągłe **kontury**, które w zależności od zastosowania, mogą być bardziej użyteczne [24]. Kontur jest listą lub posortowaną tablicą punktów krawędzi [24], ma więc strukturę uporządkowaną. Jeżeli punkt krawędzi wykryto na podstawie zmiany znaku pewnej funkcji (jak, np. opisano powyżej), operacja łączenia w ciąg sprowadza się do wybrania jednego elementu i dołączaniu jego kolejnych sąsiadów w obu kierunkach [24].

2.1.3. Proste i krzywe matematyczne

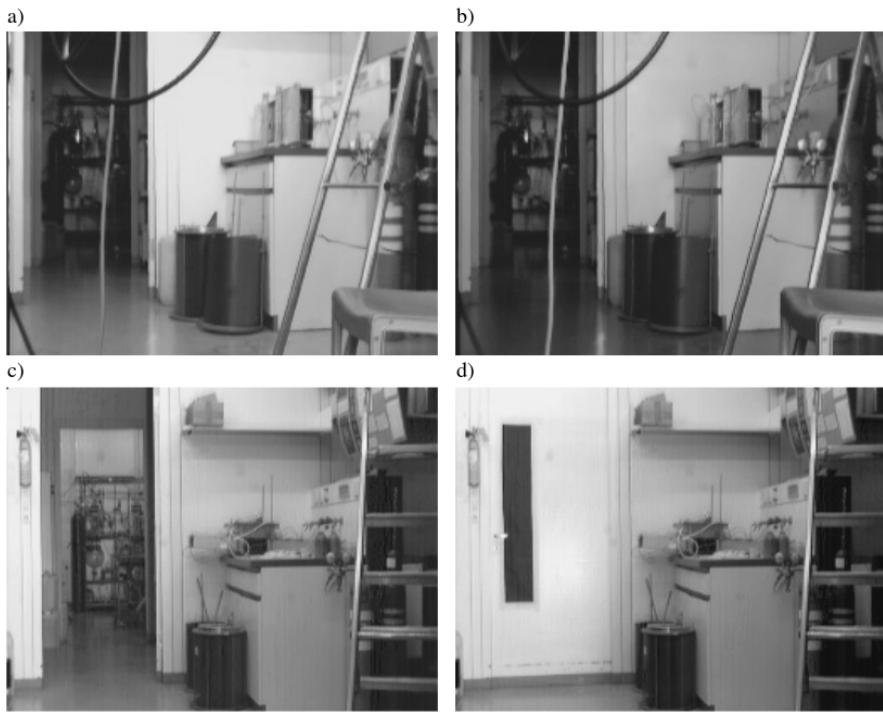
W wyniku operacji łączenia punktów krawędzi otrzymuje się reprezentację krzywej w przestrzeni 2-D w postaci należących do niej punktów [24]. Na ich podstawie można następnie wykonać aproksymację, mającą na celu uzyskanie reprezentacji w postaci **prostej lub krzywej opisanej równaniami matematycznymi**. Przy zastosowaniu odpowiednich metod (w szczególności transformaty Hougha) jest to możliwe nawet w przypadku wystąpienia na obrazie przerw lub przysłonięć konturu [24]. Same cechy o tej postaci nie są jednak użyteczne w przypadku zadania śledzenia obiektów, wobec czego zagadnienie to nie zostanie szerzej omówione.

2.2. Analiza ruchu

2.2.1. Ruch w sekwencji obrazów

Analiza ruchu obserwowanego w sekwencji obrazów jest kwintesencją algorytmów śledzenia obiektów, oraz wielu innych, jak np. kompresja video (w tym *MPEG* i *H.263*), stabilizacja obrazu, obrazowa diagnostyka medyczna, rekonstrukcja sceny 3-D na podstawie obrazu z poruszającej się kamery, etc. [24]. Ze względu na postać przetwarzanych danych - sekwencje zamiast pojedynczych obrazów - analiza ruchu jest stosunkowo wymagająca obliczeniowo oraz pamięciowo.

Intuicyjnie, opiera się ona na obserwowaniu różnic pomiędzy dwoma kolejnymi klatkami [13]. W trywialnym ujęciu, można ją zrealizować właśnie poprzez odjęcie od siebie dwóch kolejnych klatek, czyli poprzez wyznaczenie różnicy wartości pikseli o tych samych współrzędnych. Takie podejście okazuje się jednak bardzo szybko prowadzić do błędnej interpretacji, ponieważ zmiany wartości pikseli wynikają nie tylko z ruchu obiektów, ale również z występujących zmian iluminacji [13]. Mogą mieć one charakter globalny - zmienia się oświetlenie ca-



Rysunek 2.4: Przykład różnic pomiędzy kolejnymi obrazami sekwencji

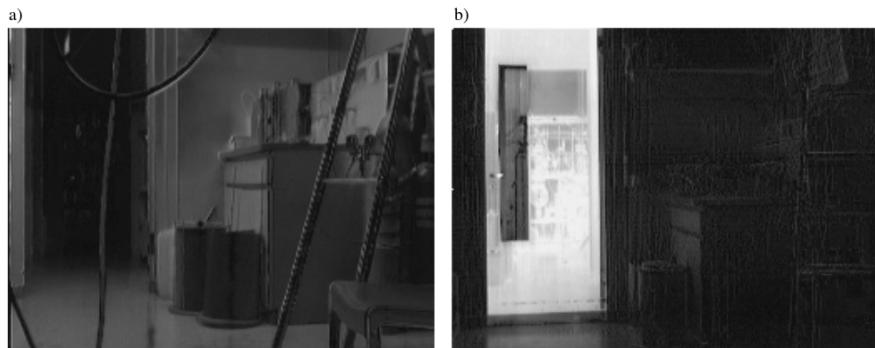
Definicja par: a) poprzedza b); c) poprzedza d)

Źródło: [13]

łej sceny, lub lokalny - przemieszczenie lub inna zmiana stanu obiektu na scenie zmienia sposób w jaki odbija on światło, tym samym zmieniając iluminację innych obiektów w jego otoczeniu. Przykładowo, na rysunku 2.4 przedstawiono dwie pary obrazów tej samej sceny a na rysunku 2.5 różnice pomiędzy obrazami tych par. W pierwszej z nich zmianie ulega oświetlenie całej sceny, zmianie ulega iluminacja całej sceny, nie występuje natomiast ruch obiektów, co objawia się ogólną, niewielką zmianą wartości pikseli. W drugiej zamknięcie drzwi, a więc ich ruch, skutkuje dużą zmianą wartości pikseli przynależących do ich obszaru oraz niewielkimi lokalnymi zmianami widocznymi na innych obiektach.

Ruch w sekwencji obrazów ujawnia się więc poprzez zmiany wartości pikseli o charakterze przestrzennym i czasowym [13]. W celu jego analizy konieczna jest wiedza o związku pomiędzy punktami na obrazie pierwotnym i następującym, tzn. zdefiniowanie wektorów przemieszczeń, co nie zawsze jest to możliwe w sposób jednoznaczny. Przykładowo, na obrazie obserwowany jest pewien region x_i (rysunek 2.6). W zależności od sposobu jego zdefiniowania, scisłe określenie korespondencji pomiędzy tym regionem a odpowiadającym mu regionem na następnym obrazie sekwencji może być możliwe (rysunek 2.6a) lub nie (rysunek 2.6b i c). Zjawisko to nosi nazwę **problemu apertury**.

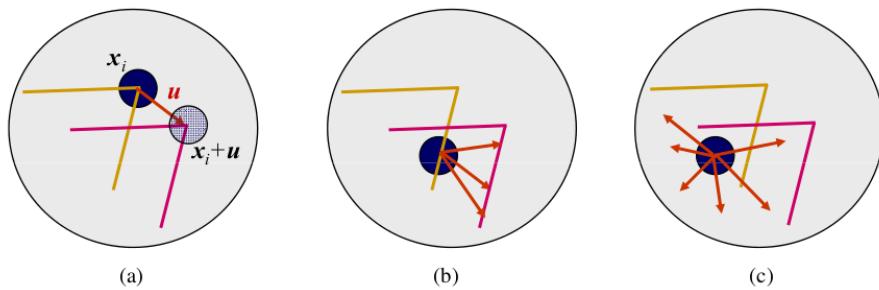
Problem apertury jest szczególnym przypadkiem **problemu korespondencji**, polegające na niemożliwości określenia korespondencji punktów na dwóch kolejnych obrazach w sekwencji [13]. Jego naturę dobrze przedstawia przykład z rysunku 2.7, na którym przedstawiono zbiór poruszających się, identycznych cząsteczek. Utworzenie powiązań pomiędzy ich pierwotnymi a wtórnymi wystąpieniami jest wykonalne, o ile odległości pomiędzy nimi są znacznie większe od długości wektorów przesunięć (rysunek 2.7a), w przeciwnym wypadku jest niemożliwe (rysunek 2.7 b)). Zjawisko to można ograniczyć zwiększąc częstotliwość próbkowania obrazów, co przekłada



Rysunek 2.5: Przykład wyniku operacji odejmowania dwóch kolejnych obrazów sekwencji

Objaśnienie: a) odejmowanie obrazów z rysunku 2.4 a) i b); b) odejmowanie obrazów z rysunku 2.4 c) i d)

Źródło: [13]



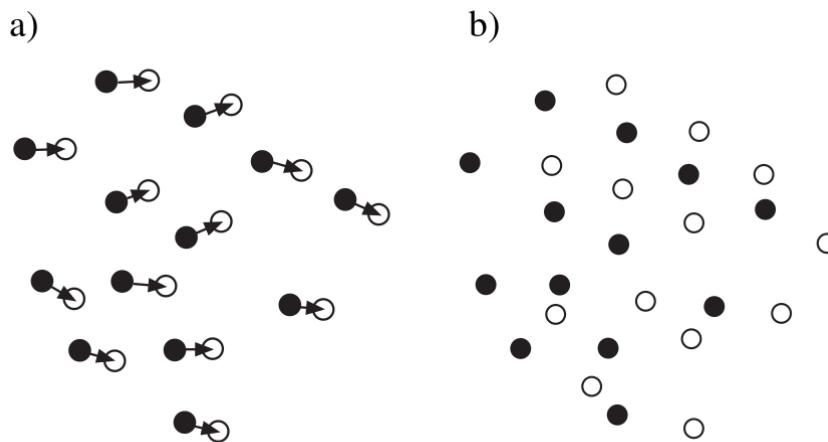
Rysunek 2.6: Warunki wystąpienia problemu apertury

Krawędź obiektu zainteresowania na obrazie pierwotnym oznaczono kolorem żółtym, na obrazie następującym kolorem fioletowym. Zdefiniowano pewien region opisywany przez współrzędne jego środka x_i , poszukiwany jest wektor przemieszczenia u , pozwalający na wyznaczenie jego kolejnych współrzędnych. Możliwe przypadki: a) wektor u może zostać wyznaczony w sposób jednoznaczny; b) ze względu na informacje zawarte w wybranym regionie x_i na obrazie następującym można wyznaczyć wiele dopasowanych do niego regionów $x_i + u$, tzn. wektor u może przyjąć wiele różnych postaci; c) informacje zawarte w regionie x_i nie narzucają żadnych wiązów dotyczących jego kolejnego położenia

Źródło: [24]

się na zmniejszenie modułów wektorów przesunięć [13]. Jego całkowita eliminacja nie jest jednak możliwa przy braku założenia minimalnej dopuszczalnej odległości między cząsteczkami.

Przedstawione powyżej przykłady obrazują podstawowy problem analizy ruchu, tzn. brak identyczności pomiędzy fizyczną korespondencją rzeczywistych obiektów a wizualną korespondencją wystającą na obrazie [13]. Określenie wizualnej korespondencji pomiędzy obiektami na dwóch obrazach jest możliwe, nawet jeżeli nie występuje pomiędzy nimi fizyczna korespondencja, natomiast fizyczna korespondencja dwóch obiektów nie musi skutkować ich wizualną korespondencją na obrazie [13].



Rysunek 2.7: Przykład występowania problemu korespondencji

Czarne koła obrazują pierwotne a białe wtórne położenia nieroóżnialnych cząsteczek. Występowanie problemu korespondencji: a) względne przesunięcia cząsteczek są małe w porównaniu do odległości między nimi, dopasowanie polega na odnalezieniu najbliższej cząsteczki; b) duże wartości modułów wektorów przesunięć uniemożliwiają wykonanie dopasowania

Źródło: [13]

2.2.2. Przepływ optyczny

W celu uchwycenia zależności pomiędzy ruchem występującym w sekwencji obrazów a zmianami wartości pikseli konieczne jest wprowadzenie dwóch pojęć - **pola ruchu** (ang. *motion field*) oraz **przepływu optycznego** (ang. *optical flow*) [13]. Pole ruchu $\mathbf{u} = [u_1, u_2]^T = [u, v]^T$ to pole wektorów rzeczywistego ruchu obiektów w przestrzeni trójwymiarowej rzutowanych na dwuwymiarową płaszczyznę obrazu, określone dla każdego jego piksela, o wymiarze prędkości [13] [3]. Przepływ optyczny $\mathbf{f} = [f_1, f_2]^T$ jest estymacją pola ruchu [24], wyznaczaną na podstawie obserwowanych przemieszczeń („przepływu”) poszczególnych pikseli obrazu rozróżnianych poprzez ich wartość [13] [3] [12]. Pole ruchu i przepływ optyczny są identyczne tylko jeśli obiekty poruszające się na scenie nie zmieniają irydancji płaszczyzny obrazu [13]. Przykładową sekwencję obrazów oraz wyznaczony dla niej przepływ optyczny przedstawiono na rysunku 2.8.

Przepływ optyczny obliczany jest dla każdego piksela obrazu. Większość metod jego wyznaczania opiera się o optymalizację pewnej funkcji globalnej energii E_G o postaci [3]:

$$E_G = E_D + \lambda E_P \quad (2.15)$$

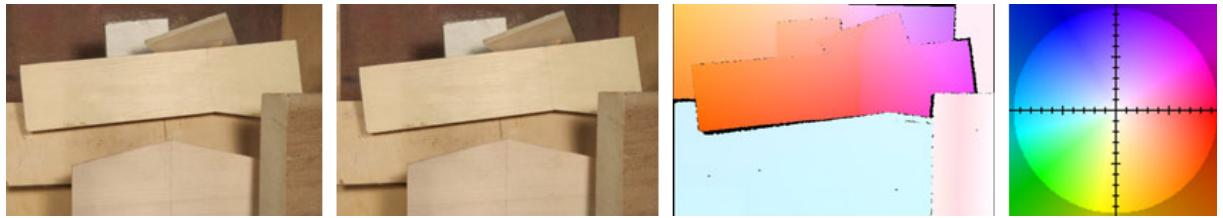
gdzie:

E_D – term danych - określenie stopnia spójności przepływu optycznego

E_P – term nadzędny - dookreślający problem poprzez preferowanie przepływów o określonym charakterze (np. płynnych)

λ – współczynnik wagowy

Podstawowa definicja termu danych opiera się o założenie **warunku stałej jasności** (ang. *brightness constancy*), tzn. że piksel odpowiadający pojedynczemu punktowi poruszającego się obiektu zmieniając położenie na



Rysunek 2.8: Przykład wizualizacji przepływu optycznego

Od lewej: dwa kolejne obrazy sekwencji, wizualizacja wyznaczonego przepływu optycznego, kodowanie modułu przepływu (jednostki na osiach mają wymiar pikseli). Pomiędzy klatkami nastąpił ruch sztywnego obiektu na pierwszym planie, tło pozostało nieruchome (brak ruchu kamery)

Źródło: [3]

obrazie nie zmienia swojej wartości lub koloru [3]. Formalną postać tego warunku przedstawia równanie 2.16.

$$I(x, y, t) = I(x + u, y + v, t + 1) \quad (2.16)$$

gdzie:

$I(x, y, t)$ – wartość piksela (x, y) na obrazie I w chwili t
 $(u(x, y, t), v(x, y, t))$ – przepływ

Rozwinięcie równania 2.16 w szereg Taylora pierwszego rzędu pozwala na wyznaczenie **równania przepływu optycznego** (ang. *optical flow constraint*):

$$u \frac{\partial I}{\partial x} + v \frac{\partial I}{\partial y} + \frac{\partial I}{\partial t} = 0 \quad (2.17)$$

Równania 2.16 i 2.17 zawierają po dwie niewiadome, elementy nieznanego przepływu $u(x, y, t), v(x, y, t)$, przy jednokrotnym narzuceniu pojedynczego ograniczenia. Oznacza to, że zagadnienie wyznaczenia przepływu jest niedookreślone, stąd konieczność wprowadzenia do równania 2.15 termu nadzorującego. Jest to również formalna przyczyna występowania wspomnianego wcześniej problemu apertury [3]. Oba równania mogą służyć do wyznaczenia postaci termu danych, przy czym ich znaczenie jest praktycznie tożsame, alternatywą stanowi założenie o wysokim współczynniku korelacji pomiędzy dwoma kolejnymi obrazami [3]. Stosowane równanie jest przekształcane do postaci, z której wylicza się wartość błędu dla każdego piksela. Wartości błędów są następnie agregowane, w wyniku czego otrzymujemy się funkcję kary, jak np. w algorytmie Horna-Schuncka [12] [3] [14]:

$$E_D = \sum_{x,y} (u \frac{\partial I}{\partial x} + v \frac{\partial I}{\partial y} + \frac{\partial I}{\partial t})^2 \quad (2.18)$$

W celu zwiększenia odporności na zmiany wyglądu sceny nie wynikające z ruchu, niektóre metody zamiast wartości pikseli używają cech, np. opisanych w podrozdziale 2.1 punktów charakterystycznych *SIFT* [3], lub innych, specjalnie do tego przystosowanych. Zwiększenie odporności można również uzyskać poprzez wprowadzenie do równania 2.16 zmiany wartości piksela wprost [3]

$$g(x, y)I(x, y, t) = I(x + u, y + v, t + 1) + b(x, y) \quad (2.19)$$

gdzie:

$g(x, y)$ – współczynnik skalujący
 $b(x, y)$ – współczynnik przesunięcia

Oczywistą konsekwencją takiego działania jest zwiększenie liczby niewiadomych do czterech, wymaga więc ono narzucenia na zmiany zmianę wartości piksela zdefiniowaną przez $g(x, y)$ i $b(x, y)$ odpowiednich więzów [3].

Term nadrzędny w równaniu 2.15 ma za zadanie wprowadzenie dodatkowych ograniczeń, umożliwiających wyznaczenie jego jednoznacznego rozwiązania. Najczęściej dotyczą one gładkości przepływu [3]. W najprostszej postaci, stosowanej w algorytmie Horna-Schuncka, preferowany jest przepływ o małym gradiencie, sam term nadrzędny przyjmuje postać [12] [3]:

$$E_P = \sum_{x,y} [(\frac{\partial u}{\partial x})^2 + (\frac{\partial u}{\partial y})^2 + (\frac{\partial v}{\partial x})^2 + (\frac{\partial v}{\partial y})^2] \quad (2.20)$$

Ulepszenie tej koncepcji polega na nadaniu składnikom sumy wag, wyznaczanych na podstawie pewnej funkcji zmieniającej wartość w zależności od przestrzennego położenia rozpatrywanego piksela. W szczególności, może ona operować na gradiencie obrazu (zmodyfikowaną postać termu nadrzędnego przedstawia równanie 2.21) i maleć wraz z jego wzrostem, co zmniejsza znaczenie pikseli należących do krawędzi. Jest to korzystne, ponieważ zaburzenie ciągłości przepływu jest bardziej prawdopodobne właśnie na krawędziach [3].

$$E_P = \sum_{x,y} w(\nabla I)[(\frac{\partial u}{\partial x})^2 + (\frac{\partial u}{\partial y})^2 + (\frac{\partial v}{\partial x})^2 + (\frac{\partial v}{\partial y})^2] \quad (2.21)$$

gdzie:

$w(\nabla I)$ – funkcja wagi

Innym możliwym założeniem służącym do wyznaczenia termu nadrzędnego jest założenie sztywności, np. preferujący przepływ przebiegający wzdłuż linii epipolarnych [3].

Oba składniki równania energii globalnej (równanie 2.15) mają charakter funkcji kary. Wyznaczenie przepływu dla danego piksela polega na rozwiązaniu zadania jej minimalizacji, w którym zmiennymi decyzyjnymi są elementy definiującego wektora (u, v) . Dwie podstawowe metody stosowane w tym celu to **metoda gradientu prostego** oraz **rachunek wariacyjny** [3].

Metoda gradientu prostego polega na iteracyjnym przybliżaniu rozwiązania optymalnego kolejnymi punktami w przestrzeni poszukiwań wyznaczanymi na podstawie gradientu funkcji celu, w tym przypadku:

$$-\nabla E_G(\mathbf{f}) = -\frac{\partial E_G}{\partial \mathbf{f}} \quad (2.22)$$

gdzie:

\mathbf{f} – wektor przepływu $\mathbf{f} = (u, v)$

W przypadku zastosowania rachunku wariacyjnego zakłada się, że energię globalną E_G można przedstawić w formie [3]:

$$E_G = \iint E(u(x, y), v(x, y), x, y, u_x, u_y, v_x, v_y) dx dy \quad (2.23)$$

gdzie:

$$\begin{aligned} u_x &= \frac{\partial u}{\partial x} \\ u_y &= \frac{\partial u}{\partial y} \\ v_x &= \frac{\partial v}{\partial x} \\ v_y &= \frac{\partial v}{\partial y} \end{aligned}$$

Składowe przepływu u i v są w tym przypadku traktowane jako funkcje położenia $u(x, y)$ i $v(x, y)$, ich parametryzacja następuje później. Minimum funkcjonału energii E_G jest wyznaczane na podstawie równań Eulera-Lagrange'a [12] [3]:

$$\frac{\partial E_G}{\partial u} - \frac{\partial}{\partial x} \frac{\partial E_G}{\partial u_x} - \frac{\partial}{\partial y} \frac{\partial E_G}{\partial u_y} = 0 \quad (2.24)$$

$$\frac{\partial E_G}{\partial v} - \frac{\partial}{\partial x} \frac{\partial E_G}{\partial v_x} - \frac{\partial}{\partial y} \frac{\partial E_G}{\partial v_y} = 0 \quad (2.25)$$

Przykładowo, dla algorytmu Horna-Schuncka E_G przyjmuje postać [12]:

$$E_G = \iint (I_x u + I_y v + I_t)^2 + \alpha^2 (u_x^2 + u_y^2 + v_x^2 + v_y^2) dx dy \quad (2.26)$$

gdzie:

$$I_x = \frac{\partial I}{\partial x}$$

$$I_y = \frac{\partial I}{\partial y}$$

$$I_t = \frac{\partial I}{\partial t}$$

α^2 – współczynnik wagi

Po podstawieniu równania 2.26 do równań 2.24 i 2.25:

$$I_x^2 u + I_x I_y v = \alpha^2 \nabla^2 u - I_x I_t \quad (2.27)$$

$$I_x I_y u + I_y^2 v = \alpha^2 \nabla^2 v - I_y I_t \quad (2.28)$$

gdzie:

∇^2 – laplasjan

Laplasjany $\nabla^2 u$ i $\nabla^2 v$ w równaniach 2.27 i 2.28 są aproksymowane przez różnicę wartości rozpatrywanego piksela $u(x, y)$ lub $v(x, y)$ oraz średniej ważonej wartości pikseli jego otoczenia $\bar{u}(x, y)$ lub $\bar{v}(x, y)$ [12]:

$$\nabla^2 u \approx \bar{u}(x, y) - u(x, y) \quad \nabla^2 v \approx \bar{v}(x, y) - v(x, y) \quad (2.29)$$

Po podstawieniu równania 2.29 do równań 2.27 i 2.28 i wykonaniu przekształceń, otrzymuje się układ dwóch równań z dwiema niewiadomymi (u, v):

$$(\alpha^2 + I_x^2 + I_y^2)(u - \bar{u}) = -I_x(I_x \bar{u} + I_y \bar{v} + I_t) \quad (2.30)$$

$$(\alpha^2 + I_x^2 + I_y^2)(v - \bar{v}) = -I_y(I_x \bar{u} + I_y \bar{v} + I_t) \quad (2.31)$$

Wyznaczenie przepływu optycznego polega na rozwiązaniu układu równań 2.30 i 2.31 dla każdego piksela obrazu. Ze względu na dużą złożoność obliczeniową takiego podejścia, lepszym rozwiązaniem jest iteracyjne wyliczanie kolejnych wartości (u_{n+1}, v_{n+1}) na podstawie estymowanych wartości pochodnych I_x , I_y i I_t oraz średniej z poprzednich wartości (u_n, v_n) , przy założeniu zerowych warunków początkowych:

$$u_{n+1} = \bar{u}_n - I_x \frac{I_x \bar{u}_n + I_y \bar{v}_n + I_t}{\alpha^2 + I_x^2 + I_y^2} \quad (2.32)$$

$$v_{n+1} = \bar{v}_n - I_y \frac{I_x \bar{u}_n + I_y \bar{v}_n + I_t}{\alpha^2 + I_x^2 + I_y^2} \quad (2.33)$$

Powyższe rozważania dotyczą metod wyznaczania przepływu optycznego dla każdego piksela obrazu, który z tego powodu określa się mianem **gęsteego** (ang. *dense*). Alternatywnym podejściem, realizowanym np. przez algorytm Lucas-Kanade-Tomasi 3.1, jest wyznaczenie przepływu optycznego w postaci **rzadkiej** (ang. *sparse*), tzn. jedynie dla wybranego zbioru punktów obrazów, np. wykrytych punktów charakterystycznych [14]. Gęsty przepływ optyczny jest z natury dokładniejszą estymacją pola ruchu, jednak jego wyznaczanie jest znacznie bardziej złożone obliczeniowo.

3. Algorytmy śledzenia obiektów w robotyce mobilnej

Filtr cząsteczkowy

W niniejszym rozdziale omówione szczegółowo zostaną algorytmy spotykane w literaturze w kontekście systemów śledzenia obiektów dla robotów mobilnych. Opisy konkretnych przykładów istniejących rozwiązań umieszczone są w rozdziale 4. Taki podział jest dogodny, ponieważ istniejące systemy często agregują różne algorytmy, lub wykorzystują ich zmodyfikowane wersje.

3.1. Algorytm Lucasa-Kanade

Jak wspomniano wspomniano w podrozdziale 2.2.2 analizę ruchu zarejestrowanego w sekwencji obrazów można przeprowadzić posługując się przepływem optycznym. Takie podejście znajduje zastosowanie powszechnie zastosowanie w systemach śledzenia wizyjnego, również w robotyce mobilnej. Szczególnie popularną metodą jego wyznaczania jest **algorytm Lucasa-Kanade**, wykorzystany w różnych jego odmianach w [17], [20] i [21].

Zastosowanie algorytmu Lucasa-Kanade wykracza znacznie poza zagadnienie śledzenia obiektów i obejmuje między innymi mozaikowanie obrazów, obrazowanie medyczne czy rozpoznawanie twarzy [2]. Sam algorytm został po raz pierwszy opublikowany w roku 1981 przez Bruce'a Lucasa oraz Takeo Kanade'a, od tego czasu w literaturze zaproponowano szereg jego modyfikacji.

3.1.1. Klasyczny algorytm Lucasa-Kanade

Pierwsza postać algorytmu pomimo swojego wieku wciąż znajduje praktyczne zastosowanie - wykorzystano ją w rozwiązaniu [21]. Oryginalną funkcją tej wersji, nazywaną czasem w literaturze addytywną [2], było dopasowanie obrazu wzorcowego $T(\mathbf{x})$ do obrazu wejściowego $I(\mathbf{x})$, gdzie $\mathbf{x} = (x, y)^T$ jest wektorem współrzędnych pikseli. W przypadku wyznaczania przepływu optycznego, obraz wzorcowy $T(\mathbf{x})$ jest podregionem obrazu z sekwencji w chwili $t = 1$, natomiast $I(\mathbf{x})$ jest obrazem w chwili $t = 2$ [2]. Zakłada się, że wzorzec $T(\mathbf{x})$ uległ na obrazie $I(\mathbf{x})$ pewnemu sparametryzowanemu odkształceniu (ang. *warp*) $\mathbf{W}(\mathbf{x}, \mathbf{p})$, gdzie \mathbf{p} jest wektorem parametrów. Odkształcenie $\mathbf{W}(\mathbf{x}, \mathbf{p})$ przyporządkowuje pikselom wzorca $T(\mathbf{x})$ ich subpikselowe położenie $W(\mathbf{x}, \mathbf{p})$ w układzie współrzędnych obrazu I [2], przykład przedstawiono na rysunku 3.1. Odkształcenie pomiędzy dwoma kolejnymi obrazami w sekwencji, dla których wyznaczany jest przepływ optyczny ma postać [2]:

$$\mathbf{W}(\mathbf{x}, \mathbf{p}) = \begin{bmatrix} x + p_1 \\ y + p_2 \end{bmatrix} \quad (3.1)$$

gdzie:

$$\mathbf{p} = (p_1, p_2)^T - \text{przepływ optyczny}$$

Zasadniczym celem algorytmu jest minimalizacja sumy kwadratów błędów pomiędzy wzorcem $T(\mathbf{x})$ a odkształconym obrazem wejściowym $I(W(\mathbf{x}, \mathbf{p}))$ [2]:

$$\sum_{\mathbf{x}} (I(W(\mathbf{x}, \mathbf{p})) - T(\mathbf{x}))^2 \quad (3.2)$$

Ponieważ przekształcenie $W(\mathbf{x}, \mathbf{p})$ zwraca wartości subpixselowe, do wyznaczenia wartości pikseli $I(W(\mathbf{x}, \mathbf{p}))$ konieczne jest wykonanie ich interpolacji. Wyznaczenie przepływu optycznego sprowadza się do minimalizacji wyrażenia 3.2, gdzie zmienną decyzyjną jest wektor \mathbf{p} . Takie zadanie optymalizacji ma charakter nieliniowy, ponieważ pomiędzy wartościami pikseli $I(\mathbf{x})$ a ich współrzednymi \mathbf{x} nie występuje zależność liniowa (ogólnie nie występuje żadna zależność) [2]. Minimalizacja jest wykonywana iteracyjnie, poprzez założenie znajomości pewnej estymacji p oraz wielokrotną zmianę jej wartości o Δp i wyznaczenie wartości błędu dla bieżącej postaci:

$$\sum_{\mathbf{x}} (I(W(\mathbf{x}, \mathbf{p} + \Delta \mathbf{p})) - T(\mathbf{x}))^2 \quad (3.3)$$

$$\mathbf{p} \leftarrow \mathbf{p} + \Delta \mathbf{p} \quad (3.4)$$

Wartość $\Delta \mathbf{p}$ jest wyznaczana przy każdej iteracji, warunkiem zatrzymania algorytmu jest zwykle spadek pewnej jej normy $\|\Delta \mathbf{p}\|$ poniżej wartości progowej ϵ [2].

Składnik $I(W(\mathbf{x}, \mathbf{p} + \Delta \mathbf{p}))$ jest linearyzowany poprzez rozwinięcie w szereg Taylora, w wyniku czego wyrażenie 3.3 przyjmuje postać [2]:

$$\sum_{\mathbf{x}} (I(W(\mathbf{x}, \mathbf{p})) + \nabla I \frac{\partial \mathbf{W}}{\partial \mathbf{p}} \Delta \mathbf{p} - T(\mathbf{x}))^2 \quad (3.5)$$

gdzie:

∇I – gradient obrazu $\nabla I = (\frac{\partial I}{\partial x}, \frac{\partial I}{\partial y})$

$\frac{\partial \mathbf{W}}{\partial \mathbf{p}}$ – macierz Jacobiego dla odkształcenia $W(\mathbf{x}, \mathbf{p})$

Gradient obrazu ∇I obliczany jest dla jego układu współrzędnych i odkształcanego zgodnie z bieżącą estymacją $W(\mathbf{x}, \mathbf{p})$.

Znalezienie \mathbf{p} , dla którego wyrażenie 3.5 przyjmuje minimalną wartość sprowadza się do rozwiązania problemu najmniejszych kwadratów metodą Gaussa-Newtona [2]. W tym celu wyznacza się jego pochodną cząstkową ze względu na $\Delta \mathbf{p}$ oraz porównuje ją do zera. Po przekształceniach:

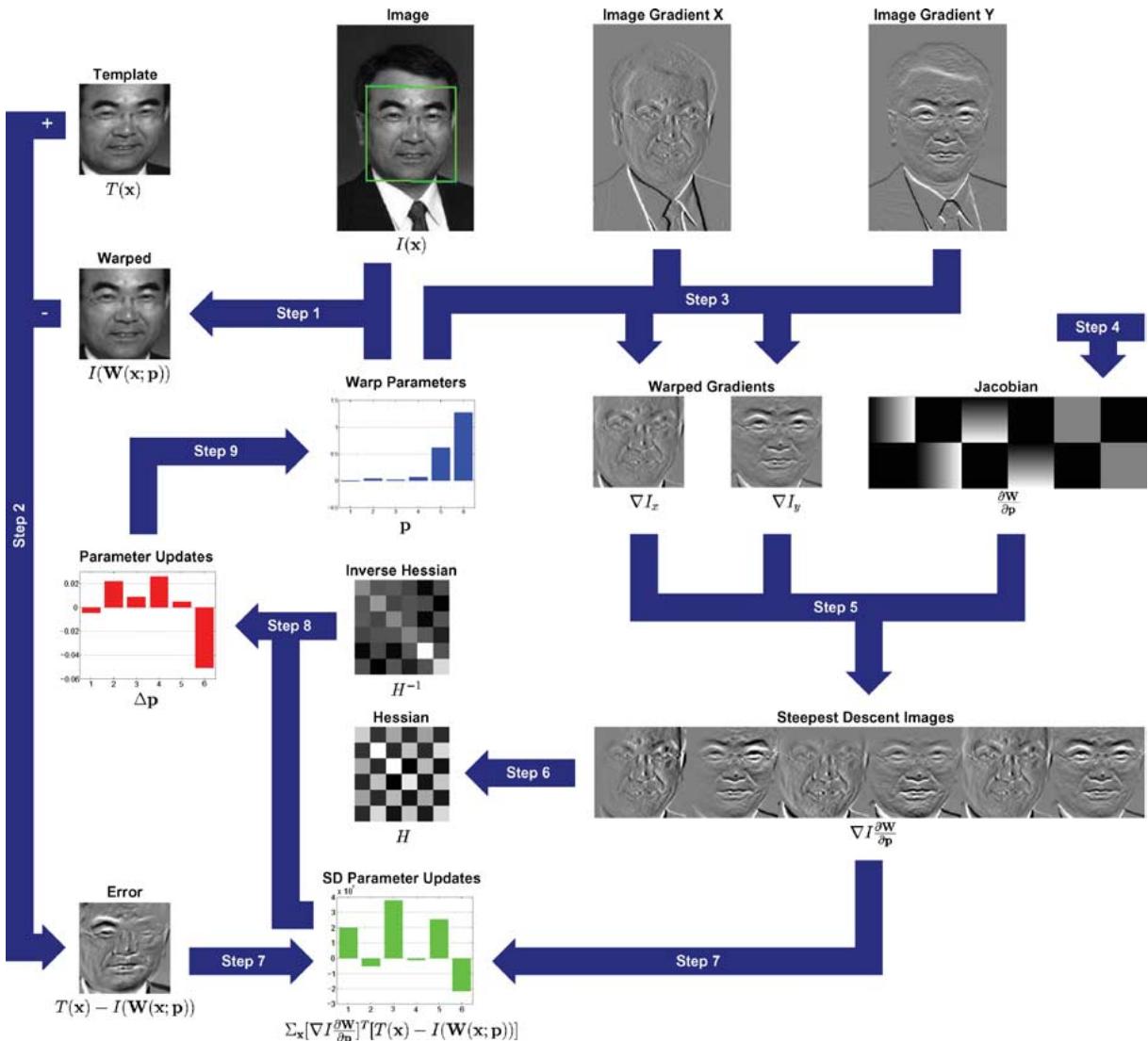
$$\Delta \mathbf{p} = H^{-1} \sum_{\mathbf{x}} (\nabla I \frac{\partial \mathbf{W}}{\partial \mathbf{p}})^T (T(\mathbf{x}) - I(W(\mathbf{x}, \mathbf{p}))) \quad (3.6)$$

$$H = \sum_{\mathbf{x}} (\nabla I \frac{\partial \mathbf{W}}{\partial \mathbf{p}})^T (\nabla I \frac{\partial \mathbf{W}}{\partial \mathbf{p}}) \quad (3.7)$$

gdzie:

H – aproksymacja hesjanu według metody Gaussa-Newtona

Cały algorytm sprowadza się do iteracyjnego wyznaczania równań 3.6 i 3.4. Realizację jego poszczególnych etapów przedstawiono na rysunku 3.1. Wykonanie pojedynczej iteracji ma złożoność obliczeniową $O(n^2 N + n^3)$, gdzie N to liczba pikseli wzorca T , a n to liczba parametrów odkształcenia [2].



Rysunek 3.1: Przebieg algorytmu Lucasa-Kanade

Krok 1 - wykonanie odkształcenia $W(\mathbf{x}, \mathbf{p})$ na obrazie wejściowym $I(\mathbf{x})$ według jego ostatniej estymacji; krok 2 - obliczenie bieżącej wartości błędu; krok 3 - wyznaczenie gradientu oraz jego odkształcenie; krok 4 - wyznaczenie macierzy Jacobiego; krok 5 - wyznaczenie $\nabla I \frac{\partial W}{\partial p}$; krok 6 - wyznaczenie hesjanu; krok 7 - obliczenie wartości wyrażenia $\sum_x (\nabla I \frac{\partial W}{\partial p})^T (T(\mathbf{x}) - I(W(\mathbf{x}, \mathbf{p})))$; krok 8 - obliczenie zmiany wartości parametrów odkształcenia $\Delta \mathbf{p}$; krok 9 - uaktualnienie wektora \mathbf{p}

Źródło: [2]

3.1.2. Algorytm Kanade-Lucasa-Tomasi

Sam algorytm Lucasa-Kanade może być w zasadzie jedynie metodą wyznaczania gęstego przepływu optycznego [28]. Z drugiej strony wiadomo, że pewne punkty obrazu lepiej nadają się do realizacji zadania śledzenia obiektu niż inne [25] - dla punktów leżących na krawędziach ujawnia się problem apertury (rozdział 2.2.1), natomiast punkty leżące wewnątrz obszarów o stałej intensywności nie dostarczają żadnej informacji o ruchu. W publikacji [25] zaproponowano metodę wyznaczania optymalnych punktów charakterystycznych oraz ich śledze-

nia z wykorzystaniem algorytmu Lucasa-Kanade, nazywaną **algorytmem Kanade-Lucas-Tomasi** (ang. *Kanade-Lucas-Tomasi tracker*, w skrócie *KLT*). Algorytm Lucasa-Kanade w tej postaci został zastosowany w [17].

Podstawę zasady działania *KLT* stanowi równanie:

$$I(x, y, t + \tau) = I(x - \xi, y - \eta, t) \quad (3.8)$$

gdzie:

$I(x, y, t)$ – wartość piksela na obrazie I o współrzędnych (x, y) w chwili t

τ – interwał czasowy pomiędzy dwoma klatkami

$\mathbf{d} = (\xi, \eta)$ – przemieszczenie punktu $\mathbf{x} = (x, y)$ pomiędzy klatkami t i $t + \tau$

Jego interpretacja jest następująca - późniejsza klatka $I(x, y, t + \tau)$ może zostać odtworzona z wcześniejszej klatki $I(x, y, t)$ poprzez odpowiednie przesunięcie jej każdego piksela, którego dla danego punktu \mathbf{x} wynosi \mathbf{d} (ogólnie jest to funkcja współrzędnych (x, y) , czasu t i interwału τ) [25]. W rzeczywistości równanie 3.8 jest rzadko prawdziwe, nawet przy statycznej scenie o stałym oświetleniu [25].

KLT w odróżnieniu od podstawowego algorytmu Lucasa-Kanade nie śledzi pojedynczych punktów, ale wybrane okna obrazu. Pozwala to zwiększenie odporności na szum oraz zmniejszenie prawdopodobieństwa błędnego rozpoznania punktu na kolejnym obrazie [25]. Z drugiej strony, nie wszystkie piksele należące do konkretnego okna muszą poruszać się w jednakowy sposób (np. jeśli okno obejmuje obszar na granicy dwóch przysłaniających się obiektów), z czym wiążą się dwa problemy: po pierwsze - jak stwierdzić, czy śledzone jest to samo okno, jeżeli jego zawartość zmienia się w czasie; po drugie - jak na podstawie informacji o prędkościach poszczególnych pikseli wnioskować o przemieszczeniu całego okna [25]. Pierwszy problem można rozwiązać sprawdzając, jak bardzo okno zmienia się pomiędzy kolejnymi klatkami, oraz, ewentualnie, odrzucając je, jeśli zmiana jest zbyt duża. Rozwiązaniem drugiego problemu jest wprowadzenie bardziej skomplikowanego modelu zmiany okna (jak np. przekształcenia afoniczne, jak w opisanej dalej modyfikacji według [22]), jednak tworzy to ryzyko nadokreślenia całego zagadnienia, ze względu na częste występowanie w sekwencjach obrazów obiektów sztywnych [25]. Z tego powodu *KLT* operuje na małych oknach, dla których wyznacza się jedynie wektor przemieszczenia, zaś każda występująca pomiędzy dwoma kolejnymiinstancjami okna rozbieżność nie opisana przez niego traktowana jest jako błąd [25]. W skutek przyjęcia założenia o małych rozmiarach okien, oryginalny algorytm *KLT* nie radzi sobie w przypadku dużych przemieszczeń punktów pomiędzy klatkami.

Przyjmując oznaczenie $J(\mathbf{x}) = I(\mathbf{x}, y, t + \tau)$, model przemieszczenia punktu pomiędzy klatkami można zdefiniować następująco:

$$J(\mathbf{x}) = I(\mathbf{x} - \mathbf{d}) + n(\mathbf{x}) \quad (3.9)$$

gdzie:

\mathbf{d} – wektor przemieszczenia punktu $\mathbf{d} = (\xi, \eta)$

$I(\mathbf{x})$ – Wartość piksela na obrazie I o współrzędnych \mathbf{x} w chwili t

$n(\mathbf{x})$ – szum

Z równania 3.9 należy wyznaczyć wektor przemieszczenia \mathbf{d} tak, aby zminimalizować błąd dopasowania:

$$\epsilon = \int_W (I(\mathbf{x} - \mathbf{d}) - J(\mathbf{x}))^2 w \, d\mathbf{x} \quad (3.10)$$

gdzie:

w – współczynnik wagowy

Współczynnik wagowy w może być stały i wynosić 1 lub może przyjmować wartości zgodnie z rozkładem Gaussa, w celu podkreślenia wagi centralnej części okna [25]. Jak widać postać równania 3.10 jest analogiczna do bazowego równania klasycznego algorytmu Lucasa-Kanade 3.2. Całe zadanie śledzenia jest wykonywane poprzez jego zastosowanie dla punktów reprezentujących środki wybranych okien.

W algorytmie *KLT* zdefiniowano również metodę doboru okien dobrze nadających się do zastosowania w zadaniu śledzenia. Opiera się ona na formalnym powiązaniu właściwości macierzy H z równania 3.6 dla danego punktu z właściwościami jego otoczenia [25]. Po pierwsze, z równania 3.6 wynika wprost, że aby śledzenie było możliwe macierz H musi być niezdegenerowana, w związku z czym jej wartości własne λ_1 i λ_2 muszą być większe od poziomu szumu występującego na obrazie [25]. Po drugie, całe zadanie powinno być dobrze uwarunkowane, co oznacza, że wartości własne λ_1 i λ_2 nie powinny znacznie się od siebie różnić [25].

Istnieją trzy możliwe przypadki wzajemnych relacji wartości własnych macierzy H :

1. $\lambda_1 \gg \lambda_2$ lub $\lambda_1 \ll \lambda_2$ - otoczenie punktu stanowi wzór jednokierunkowy (krawędź) i nie jest odpowiednie do śledzenia, ze względu na problem apertury (rozdział 2.2).
2. $\lambda_1 \approx \lambda_2 \approx 0$ - otoczenie punktu jest obszarem o stałej intensywności i nie jest odpowiednie do śledzenia, ponieważ nie przechowuje informacji o ruchu
3. $\lambda_1 \approx \lambda_2 \gg 0$ - otoczenie punktu jest zróżnicowane teksturowo (narożniki, wzór pieprz-sól) i dobrze nadaje się do zadania śledzenia

W praktyce, jeżeli mniejsza wartość własna spełnia założenie o przewyższaniu poziomu szumu, cała macierz H jest również dobrze uwarunkowana, co wynika z faktu ograniczenia wartości pikseli do dozwolonego przedziału [25]. Warunek służący do poszukiwania okien odpowiednich do śledzenia można zdefiniować następująco:

$$\min(\lambda_1, \lambda_2) > \lambda \quad (3.11)$$

gdzie:

λ – wartość progowa

Wartość progową λ można wyznaczyć poprzez uśrednienie wyników pomiarów wartości własnych macierzy H dla arbitralnie wybranych obszarów o stałej jasność (dolina granica) oraz obszarów zróżnicowanych teksturowo (górska granica) [25].

W publikacji [22] przedstawiono ulepszenie *KLT* polegające wprowadzeniu dodatkowego modelu zmiany śledzonego okna pomiędzy klatkami w postaci przekształcenia afanicznego (oryginalnie *KLT* zakłada wyłącznie czystą translację). Formalnie definicja takiego modelu polega na zastąpieniu w podstawowym równaniu 3.8 wektora przesunięcia \mathbf{d} , stanowiącego czystą translację, wektorem przesunięcia δ (określonym jako afaniczne pole ruchu) [22]

$$\delta = \mathbf{D}\mathbf{x} + \mathbf{d} \quad (3.12)$$

$$\mathbf{D} = \begin{bmatrix} d_{xx} & d_{xy} \\ d_{yx} & d_{yy} \end{bmatrix} \quad (3.13)$$

gdzie:

D – macierz deformacji

d – wektor przesunięcia środka okna

Zależność pomiędzy kolejną klatką J a poprzednią klatką I ma postać:

$$J(\mathbf{Ax} + \mathbf{d}) = I(\mathbf{x}) \quad (3.14)$$

Zadanie śledzenia polega na wyznaczeniu sześciu nieznanych parametrów, tj. elementów wektora **d** i macierzy **D**, przy czym jakość ich oszacowania jest zależna od założonej wielkości okna, zróżnicowania teksturowego jego zawartości oraz rozbieżności pomiędzy klatkami (gwałtowności występującego pomiędzy nimi ruchu) [22]. Im mniejsze okno, tym mniej miarodajna jest wyznaczona postać macierzy (D) (jej wyznaczenie jest trudniejsze przy mniejszych wariancach ruchu), jednocześnie tym lepiej nadaje się ono do śledzenia (mniej podatne na gwałtowne zmiany zawartości [25] [22]). Śledzenie jest w związku z tym realizowane, identycznie jak w wyjściowej wersji *KLT*, przy założeniu modelu ruchu w postaci czystej translacji, natomiast model ruchu w postaci przekształcenia afanicznego jest wykorzystywany do monitorowania jakości okna, poprzez porównanie jego postaci na pierwszej i bieżącej klatce [22].

3.1.3. Piramidowa wersja algorytmu Lucasa-Kanade

Wielkość śledzonego okna wpływa na dokładność i odporność algorytmu *KLT* - jak wspomniano wcześniej, mniejsze okno jest mniej podatne na zachodzące w sekwencji obrazów zmiany jasności należących do niego pikseli, pozwala więc na osiągnięcie większej dokładności. Z drugiej strony zwiększenie rozmiaru okna pozwala na śledzenie bardziej gwałtownego ruchu (większych przesunięć). Rozwiążanie tego dylematu zaproponowano w [4] - polega ona na reprezentacji obrazu w przestrzeni skali (podobnie jak w algorytmie *SIFT*, 2.1.1), nazywanej **piramidą obrazów** (ang. *image pyramid*). Takie podejście wykorzystano w rozwiązaniu [20].

Obraz wejściowy I ma wymiary $n_x \times n_y$, przyjęto notację, w której I^L oznacza L -ty poziom piramidy o wymiarach n_x^L i n_y^L , poziom zerowy to obraz nieprzeskalowany ($n_x^0 = n_x$ i $n_y^0 = n_y$). Reprezentacja piramidowa jest wyznaczana rekurencyjnie [4]:

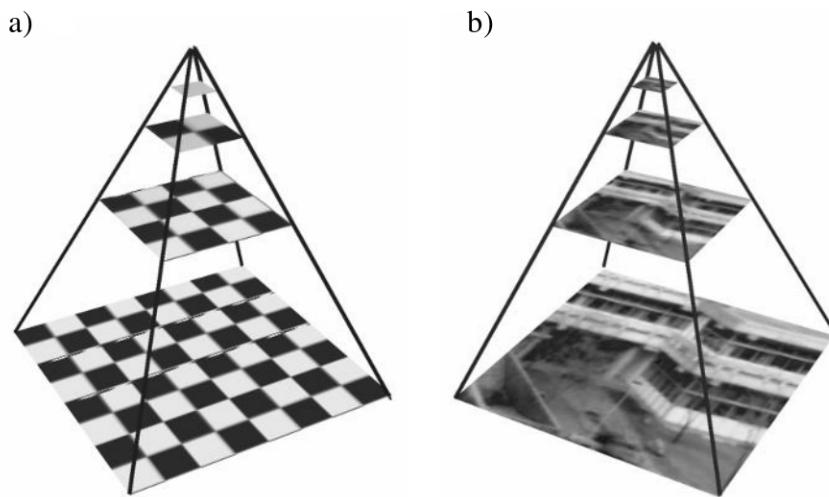
$$\begin{aligned} I^L(x, y) = & \frac{1}{4}I^{L-1}(2x, 2y) + \frac{1}{8}(I^{L-1}(2x-1, 2y) + I^{L-1}(2x+1, 2y) + I^{L-1}(2x, 2y-1) + I^{L-1}(2x, 2y+1)) \\ & + \frac{1}{16}(I^{L-1}(2x-1, 2y-1) + I^{L-1}(2x+1, 2y+1) + I^{L-1}(2x-1, 2y+1) + I^{L-1}(2x+1, 2y-1)) \end{aligned} \quad (3.15)$$

Równanie 3.15 wymaga ekstrapolacji wartości skrajnych pikseli (o jednej współrzędnej równej zero) na piksele leżące poza obrazem (o jednej współrzędnej równej -1). Rozmiar obrazu na poziomie L określany jest przez największe liczbowe całkowite n_x^L i n_y^L spełniające zależności [4]:

$$n_x^L \leq \frac{n_x^{L-1} + 1}{2} \quad (3.16)$$

$$n_y^L \leq \frac{n_y^{L-1} + 1}{2} \quad (3.17)$$

Wysokość piramidy L_m dobierana jest heurystycznie, przeważnie liczba poziomów nie przekracza 4 [4]. W równaniu 3.15 zaszyto filtr dolnoprzepustowy o masce:



Rysunek 3.2: Przykłady piramidy obrazów

Tworzenie piramidy polega na zastosowaniu rekursywnego filtrowania obrazu filtrem dolnoprzepustowym (wygładzającym) oraz jego podpróbkowaniu. Powyższe piramidy utworzono z wykorzystaniem filtra Gaussa. Oznaczenia: a) piramida Gaussa dla obrazu w postaci syntetycznego wzoru (szachownica), im wyższy poziom, tym bardziej ujawnia się przekłamanie początkowych wartości pikseli wnoszone przez filtrację; b) Piramida Gaussa utworzona dla rzeczywistego obrazu (zdjęcie fasady budynku)

Źródło: [13]

$$K = \begin{bmatrix} 1/4 & 1/2 & 1/4 \end{bmatrix} \times \begin{bmatrix} 1/4 \\ 1/2 \\ 1/4 \end{bmatrix} \quad (3.18)$$

W praktyce stosuje się większe rozmiary maski [4]. Wynikiem końcowym procedury jest ciąg obrazów, z których każdy jest dwukrotnie mniejszy od poprzedniego, podobny przypadek przedstawiono na rysunku 3.2.

Zasadniczym celem śledzenia jest wyznaczenie dla punktu o współrzędnych \mathbf{u} leżącego na obrazie I odpowiadających mu współrzędnych \mathbf{v} na kolejnym obrazie J , tzn. wyznaczenie wektora przemieszczeń \mathbf{d} , takiego że $\mathbf{v} = \mathbf{u} + \mathbf{d}$. Dla kolejnych poziomów piramidy obrazów $L = 0, \dots, L_m$ współrzędne punktu $\mathbf{u}^L = (u_x^L, u_y^L)$ wyznaczane są z zależności [4]:

$$\mathbf{u}^L = \frac{\mathbf{u}}{2^L} \quad (3.19)$$

Proces śledzenia w przestrzeni skali rozpoczyna się od wyznaczenia przepływu optycznego dla najwyższego poziomu piramidy L_m . Jest on następnie propagowany jako początkowe przybliżenie do niższego poziomu $L_m - 1$. Na podstawie początkowego przybliżenia jest wyznaczany właściwy przepływ optyczny dla poziomu $L_m - 1$, wynik jest propagowany do poziomu $L_m - 2$, itd. Rekursja kończy się po osiągnięciu poziomu 0 [4].

Zakładając znajomość początkowego przybliżenia przepływu optycznego $\mathbf{g}^L = (g_x^L, g_y^L)$ dla poziomu L , na podstawie obliczeń od poziomu L_m do $L + 1$, w celu wyznaczenia właściwego przepływu optycznego należy znaleźć wektor $\mathbf{d}^L = (d_x^L, d_y^L)$, określany jako residualny przepływ optyczny, minimalizujący funkcję błędu [4]

$$\epsilon^L(\mathbf{d}^L) = \sum_{x=u_x^L - \omega_x}^{u_x^L + \omega_x} \sum_{y=u_y^L - \omega_y}^{u_y^L + \omega_y} (I^L(x, y) - J^L(x + g_x^L + d_x^L, y + g_y^L + d_y^L))^2 \quad (3.20)$$

Zawarte w równaniu 3.20 liczby całkowite ω_x i ω_y określają rozmiar okna równy $(2\omega_x + 1) \times 2\omega_y + 1$, w obrębie którego określa się podobieństwo dla punktów $I(\mathbf{u})$ i $J(\mathbf{v})$. Wektor \mathbf{d} wyznacza się z zastosowaniem standardowego algorytmu Lucasa-Kanade 3.1.1 [4]. Następnie wynik jest propagowany do kolejnego poziomu $L - 1$, kolejne przybliżenie początkowe \mathbf{g}^{L-1} wyznacza się ze wzoru [4]:

$$\mathbf{g}^{L-1} = 2(\mathbf{g}^L + \mathbf{d}^L) \quad (3.21)$$

Dla poziomu $L - 1$ wyznaczany jest optymalny wektor \mathbf{d}^{L-1} minimalizujący funkcjonal $\epsilon^{L-1}(\mathbf{d}^{L-1})$, następuje jego propagacja do kolejnego poziomu $L - 2$ itd., aż do osiągnięcia poziomu $L = 0$. W pierwszym kroku algorytmu, w którym przetwarzany jest poziom L_m jako przybliżenie początkowe \mathbf{g}^{L_m} przyjmuje się wektor zerowy [4]. Po wykonaniu obliczeń dla wszystkich poziomów piramidy otrzymuje się rozwiązanie końcowe:

$$\mathbf{d} = \mathbf{g}^0 + \mathbf{d}^0 = \sum_{L=0}^{L_m} 2^L \mathbf{d}^L \quad (3.22)$$

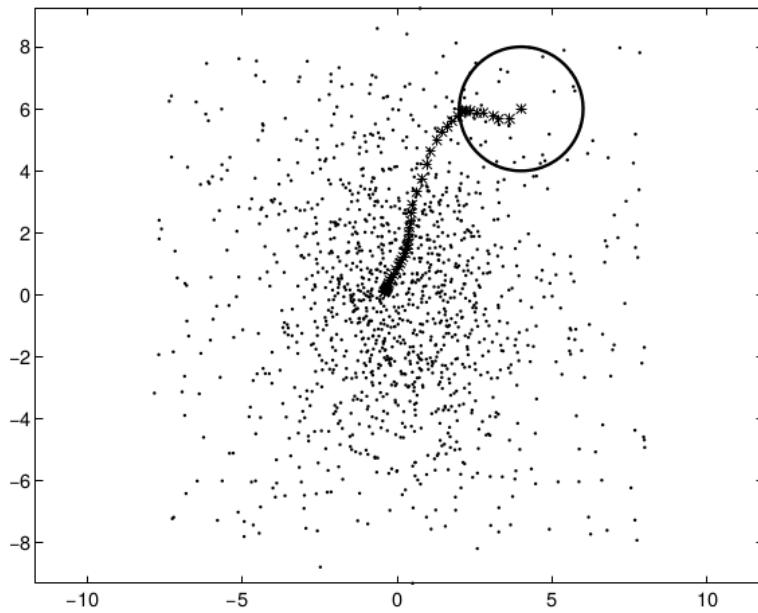
Sens piramidowej wersji algorytmu Lucasa-Kanade leży w dekompozycji właściwego przepływu optycznego dla danego poziomu piramidy na jego znane przybliżenie \mathbf{g}^L oraz część residualną \mathbf{d}^L o niewielkich wartościach, które nie wzrastają przy propagacji na kolejne poziomy. Pozwala to z powodzeniem wyznaczać ich postać z wykorzystaniem klasycznego algorytmu Lucasa-Kanade, co ostatecznie skutkuje możliwością śledzenia dużych przemieszczeń \mathbf{d} . Zakładając, że algorytm Lucasa-Kanade operujący na poszczególnych poziomach piramidy może poprawnie śledzić na każdym z nich przemieszczenie nie większe od d_{max} , ogólne maksymalne przemieszczenie dla którego wersja piramidowa działa poprawnie wynosi $d_{maxfinal} = (2^{L_m+1} - 1)$, co przy np. liczbie poziomów piramidy $L_m = 3$ daje piętnastokrotny wzrost [4].

3.2. Algorytm Mean Shift

Mean Shift to iteracyjna metoda wyznaczania ekstremów lokalnych funkcji gęstości przez estymację jej gradienit przedstawiona w roku 1975 przez Keinosuke Fukunagę oraz Larry'ego Hostetlera [9]. W przetwarzaniu obrazów, poza śledzeniem obiektów, znajduje ona zastosowanie m.in. technikach filtracji i segmentacji. W uproszczeniu, zasada działania metody *Mean Shift* opiera się na iteracyjnym przesuwaniu okna o stałym rozmiarze w kierunku średniej z próbek znajdujących się w jego wnętrzu, co wizualizuje rysunek 3.3.

3.2.1. Algorytm Mean Shift w śledzeniu obiektów

Poniżej przedstawiono zmodyfikowaną postać algorytmu *Mean Shift* zaadaptowaną do realizacji wizyjnego śledzenia obiektów, z którego korzysta rozwiązanie przedstawione w [17]. W celu realizacji zadania śledzenia należy określić reprezentację obiektu zainteresowania odpowiednią dla algorytmu, tj. jego reprezentację w postaci funkcji gęstości prawdopodobieństwa (ang. *probability density function*, w skrócie *pdf*) w wybranej przestrzeni cech (ang. *feature space*) oznaczonej jako q . Jako przestrzeń cech można przyjąć przestrzeń koloru a jako estymację *pdf* histogram obiektu wyznaczony na podstawie obrazu, wtedy [10]:



Rysunek 3.3: Przykłady działania metody *Mean Shift*

Kolejne iteracje oznaczono estymacją położenia maksimum lokalnego funkcji gęstości oznaczono symbolem "*", okno w położeniu początkowym oznaczono okręgiem

Źródło: [9]

$$\hat{\mathbf{q}} = \{\hat{q}_u\}_{u=1 \dots m} \quad \sum_{u=1}^m \hat{q}_u = 1 \quad (3.23)$$

$$\hat{\mathbf{p}}(\mathbf{y}) = \{\hat{p}_u(\mathbf{y})\}_{u=1 \dots m} \quad \sum_{u=1}^m \hat{p}_u = 1 \quad (3.24)$$

gdzie:

- $\hat{\mathbf{q}}$ – model śledzonego obiektu w postaci histogramu
- \hat{q}_u – wartość estymaty *pdf* obiektu dla przedziału klasowego u
- $\hat{\mathbf{p}}(\mathbf{y})$ – model kandydata na śledzony obiekt o współrzędnych \mathbf{y} w następnej klatce
- \mathbf{y} – wektor współrzędnych kandydata na obrazie
- $\hat{p}_u(\mathbf{y})$ – wartość estymaty *pdf* kandydata o współrzędnych \mathbf{y} dla przedziału klasowego u
- m – liczba przedziałów klasowych histogramu

Obiekt $\hat{\mathbf{q}}$ oznaczony na pierwszej klatce sekwencji jest poszukiwany na kolejnej poprzez maksymalizację pewnej funkcji podobieństwa pomiędzy nim a kandydatem $\hat{\mathbf{p}}(\mathbf{y}) \equiv \rho(\hat{\mathbf{p}}(\mathbf{y}), \hat{\mathbf{q}})$ [10]. Na obrazie śledzony obiekt reprezentowany jest elipsoidalnym obszarem (bez utraty ogólności zakłada się, że jego środek leży w punkcie 0), w celu zniesienia wpływu różnic w jego wymiarach dokonuje się normalizacji przekształcającej go do okręgu jednostkowego [10]. W celu zapewnienia regularnego przebiegu funkcji podobieństwa (ułatwiającego zadanie optymalizacji), na reprezentację obiektu nakładana jest izotropowe, wypukłe i monotonicznie malejące jądro k , przypisujące pikselom wagę malejącą wraz z oddalaniem się od środka obszaru zainteresowania. Jego zastosowanie zwiększa również odporność całego algorytmu, ponieważ piksele leżące blisko granicy obszaru zainteresowania wnoszą

najmniej wiarygodną informację (często są przysłaniane lub wpływa na nie tło) [10]. Oznaczając zbiór znormalizowanych współrzędnych pikseli leżących wewnątrz obszaru zainteresowania jako $\{\mathbf{x}_i^*\}_{i=1 \dots n}$ oraz definiując funkcję $b : R^2 \mapsto 1 \dots m$ przyporządkowującą pikselom indeks przedziału klasowego histogramu skwantowanej przestrzeni cech (koloru), prawdopodobieństwo wystąpienia konkretnej cechy u śledzonego obiektu oblicza się jako:

$$\hat{q}_u = C \sum_{i=1}^n k(\|\mathbf{x}_i^*\|^2) \delta[b(\mathbf{x}_i^*) - u] \quad (3.25)$$

$$C = \frac{1}{\sum_{i=1}^n k(\|\mathbf{x}_i^*\|^2)} \quad (3.26)$$

gdzie:

C – stała normalizacji

δ – delta Kroneckera

Wzór 3.25 pozwala na skonstruowanie modelu śledzonego obiektu w postaci histogramu. Oznaczając zbiór znormalizowanych współrzędnych pikseli należących do obszaru zainteresowania kandydata o środku w punkcie \mathbf{y} kolejnej klatki jako $\{\mathbf{x}_i\}_{i=1 \dots n_h}$, przyjmując normalizację identyczną jak w przypadku klatki zawierającej model śledzonego obiektu i stosując poprzednie jądro k ze zdefiniowanym parametrem wygładzania h , prawdopodobieństwo wystąpienia cechy u kandydata określone jest wzorem [10]:

$$\hat{p}_u(\mathbf{y}) = C_h \sum_{i=1}^{n_h} k\left(\left\|\frac{\mathbf{y} - \mathbf{x}_i}{h}\right\|^2\right) \delta[b(\mathbf{x}_i) - u] \quad (3.27)$$

$$C_h = \frac{1}{\sum_{i=1}^{n_h} k\left(\left\|\frac{\mathbf{y} - \mathbf{x}_i}{h}\right\|^2\right)} \quad (3.28)$$

gdzie:

C_h – stała normalizacji

Podstawę porównania obiektu zainteresowania i kandydata na kolejnej klatce jest funkcja podobieństwa ρ . Na jej podstawie określa się pomiędzy nimi odległość d (o charakterze metryki) [10]:

$$d(\mathbf{y}) = \sqrt{1 - \rho(\hat{\mathbf{p}}(\mathbf{y}), \hat{\mathbf{q}})} \quad (3.29)$$

Jako funkcję podobieństwa przyjmuje się estymację współczynnika Bhattacharyya pomiędzy rozkładami (modelami) \mathbf{p} i \mathbf{q} [10]:

$$\rho(\hat{\mathbf{p}}(\mathbf{y}), \hat{\mathbf{q}}) = \sum_{u=1}^m \sqrt{\hat{p}_u(\mathbf{y}) \hat{q}_u} \quad (3.30)$$

Śledzenie polega na zlokalizowaniu instancji obiektu zainteresowania na kolejnej klatce sekwencji, co jest realizowane poprzez minimalizację odległości d przy zmiennej decyzyjnej w postaci współrzędnych środka obszaru reprezentującego kandydata \mathbf{y} . Procedura wyznaczania nowego położenia rozpoczęta się w punkcie startowym stanowiącym położenie obiektu na poprzedniej klatce $\hat{\mathbf{y}}_0$ i jest przeprowadzana w oparciu o gradient estymatora jądrowego gęstości (ang. *kernel density estimator*) [10]. Minimalizacja dystansu 3.29 jest równoznaczna z maksymalizacją wartości współczynnika Bhattacharyya 3.30. Jego wartość można liniowo aproksymować stosując rozwinięcie w szereg Taylora w punkcie $\hat{p}_u(\hat{\mathbf{y}}_0)$, po przekształcenach oraz podstawieniu wzoru 3.27 [10]:

$$\rho(\hat{\mathbf{p}}(\mathbf{y}), \hat{\mathbf{q}}) \approx \frac{1}{2} \sum_{u=1}^m \sqrt{\hat{p}_u(\hat{\mathbf{y}}_0)} \hat{q}_u + \frac{C_h}{2} \sum_{i=1}^{n_h} w_i k \left(\left\| \frac{\mathbf{y} - \mathbf{x}_i}{h} \right\|^2 \right) \quad (3.31)$$

$$w_i = \sum_{u=1}^m \sqrt{\frac{\hat{q}_u}{\hat{p}_u(\hat{\mathbf{y}}_0)}} \delta[b(\mathbf{x}_i) - u] \quad (3.32)$$

W równaniu 3.31 pierwszy term nie zależy od położenia \mathbf{y} , natomiast drugi reprezentuje estymację gęstości z danym jądrem k w punkcie \mathbf{y} na bieżącej klatce, przy narzuconych współczynnikach wagowych w_i i znalezienie jego maksimum stanowi cel zadania optymalizacji [10]. W tym celu jądro jest rekurencyjnie przesuwane z bieżącego położenia $\hat{\mathbf{y}}_0$ do nowego położenia $\hat{\mathbf{y}}_1$, zgodnie ze wzorem [10]:

$$\hat{\mathbf{y}}_1 = \frac{\sum_{i=1}^{n_h} \mathbf{x}_i w_i g\left(\left\| \frac{\hat{\mathbf{y}}_0 - \mathbf{x}_i}{h} \right\|^2\right)}{\sum_{i=1}^{n_h} w_i g\left(\left\| \frac{\hat{\mathbf{y}}_0 - \mathbf{x}_i}{h} \right\|^2\right)} \quad (3.33)$$

gdzie:

$$g(x) = -k'(x)$$

Przy danym modelu obiektu $\{\hat{q}_u\}_{u=1\dots m}$ i jego położeniu na poprzedniej klatce $\hat{\mathbf{y}}_0$ cała procedura poszukiwania kandydata na kolejnej klatce przebiega następująco [10]:

1. Przyjęcie położenia kandydata na nowej klatce jako $\hat{\mathbf{y}}_0$, wyznaczenie jego modelu w tym położeniu $\{\hat{p}_u(\hat{\mathbf{y}}_0)\}_{u=1\dots m}$ i wyznaczenie w tym położeniu wartości funkcji podobieństwa $\rho(\hat{\mathbf{p}}(\hat{\mathbf{y}}_0), \hat{\mathbf{q}})$ według równania 3.30
2. Wyznaczenie wag $\{w_i\}_{i=1\dots n_h}$ zgodnie z równaniem 3.32
3. Wyznaczenie nowego położenia kandydata $\hat{\mathbf{y}}_1$ według równania 3.33
4. Wyznaczenie modelu kandydata w nowym położeniu $\{\hat{p}_u(\hat{\mathbf{y}}_1)\}_{u=1\dots m}$ i wartości funkcji podobieństwa w nowym położeniu $\rho(\hat{\mathbf{p}}(\hat{\mathbf{y}}_1), \hat{\mathbf{q}})$ według równania 3.30
5. Dopóki zachodzi warunek $\rho(\hat{\mathbf{p}}(\hat{\mathbf{y}}_1), \hat{\mathbf{q}}) < \rho(\hat{\mathbf{p}}(\hat{\mathbf{y}}_0), \hat{\mathbf{q}})$, przyjęcie nowego położenia $\hat{\mathbf{y}}_1 \leftarrow \frac{1}{2}(\hat{\mathbf{y}}_0 + \hat{\mathbf{y}}_1)$ i ponowne wyznaczenie wartości funkcji podobieństwa $\rho(\hat{\mathbf{p}}(\hat{\mathbf{y}}_1), \hat{\mathbf{q}})$
6. Jeżeli spełniony jest warunek stopu $\|\hat{\mathbf{y}}_1 - \hat{\mathbf{y}}_0\| < \epsilon$ (gdzie ϵ oznacza wartość progową) zakończenie, w przeciwnym razie podstawienie $\hat{\mathbf{y}}_0 \leftarrow \hat{\mathbf{y}}_1$ i powrót do punktu 2

W praktyce przytoczony powyżej algorytm można bardzo uprościć. Punkt 5 ma na celu jedynie uniknięcie bardzo rzadko występujących problemów numerycznych, w związku z czym można pominąć jego wykonanie, co pociąga za sobą brak konieczności realizacji punktów 1 i 4, w których wyznacza się współczynnik Bhattacharyya [10]. Jest on obliczany jednorazowo po zakończeniu działania algorytmu, w celu porównania podobieństwa modelu obiektu i wybranego kandydata [10]. Dodatkowo zastosowanie jądra Epanecznikowa o postaci danej równaniem 3.35 pozwala na znaczne zredukowanie równania 3.33 [10]:

$$\hat{\mathbf{y}}_1 = \frac{\sum_{i=1}^{n_h} \mathbf{x}_i w_i}{\sum_{i=1}^{n_h} w_i} \quad (3.34)$$

$$k(x) = \begin{cases} \frac{1}{2} c_d^{-1} (d+2)(1-x) & \text{dla } x \leq 1 \\ 0 & \text{dla } x > 1 \end{cases} \quad (3.35)$$

3.2.2. Algorytm CAMSHIFT

W kontekście wizyjnego śledzenia obiektów główną wadę algorytmu *Mean Shift* stanowi zakładana w nim stały rozwijanie rozmiaru stosowanego regionu zainteresowania - jest oczywiste, że w trakcie ruchu obiektu jego rozmiar na obrazie może ulegać zmianie. Propozycją jej eliminacji jest usprawniona wersja algorytmu o nazwie ***Continuous Adaptive Mean Shift*** (w skrócie **CAMSHIFT**), zaproponowana w [5], zastosowana w [29].

Przebieg algorytmu *CAMSHIFT* jest podobny do przedstawionej w sekcji 3.2.1 procedury *Mean Shift* i przebiega następująco [5]:

1. Wybór początkowego rozmiaru oraz położenia okna poszukiwań
2. Obliczenie położenia maksimum wewnątrz okna poszukiwań
3. Przesunięcie środka okna do położenia maksimum
4. Powrót do punktu 2 lub zakończenie w przypadku osiągnięcia zbieżności

Metoda zakłada wyznaczanie *pdf* dla wybranego obszaru obrazu, większego od okna poszukiwań, jednak w ogólnym przypadku mniejszego od całego obrazu, oraz konstrukcję histogramu w oparciu o kanał H reprezentacji obrazu w przestrzeni barw *HSV* (skrót od ang. *Hue Saturation Value*, kanał H odpowiada za reprezentację barwy piksela) [5].

Do wyznaczenia położenia maksimum (x_c, y_c) wewnątrz okna stosowane są momenty statystyczne zerowego i pierwszego rzędu, obliczane według wzorów [5]:

$$M_{00} = \sum_x \sum_y P(x, y) \quad (3.36)$$

$$M_{10} = \sum_x \sum_y xP(x, y) \quad (3.37)$$

$$M_{10} = \sum_x \sum_y yP(x, y) \quad (3.38)$$

$$x_c = \frac{M_{10}}{M_{00}} \quad (3.39)$$

$$y_c = \frac{M_{01}}{M_{00}} \quad (3.40)$$

gdzie:

x, y – współrzędne pikseli leżących wewnątrz okna poszukiwań

$P(x, y)$ – wartość rozkładu prawdopodobieństwa dla współrzędnych (x, y) określana na podstawie histogramu

M_{00} – moment rzędu zerowego

M_{10} – moment pierwszego rzędu względem osi x

M_{01} – moment pierwszego rzędu względem osi y

Z wykorzystaniem momentów drugiego rzędu M_{20}, M_{11} i M_{02} można wyznaczyć orientację θ , długość l oraz szerokość w śledzonego obiektu [5]:

$$M_{20} = \sum_x \sum_y x^2 P(x, y) \quad (3.41)$$

$$M_{11} = \sum_x \sum_y xy P(x, y) \quad (3.42)$$

$$M_{10} = \sum_x \sum_y y^2 P(x, y) \quad (3.43)$$

$$\theta = \frac{1}{2} \arctan \left(\frac{2 \left(\frac{M_{11}}{M_{00}} - x_c y_c \right)}{\left(\frac{M_{20}}{M_{00}} - x_c^2 \right) - \left(\frac{M_{02}}{M_{00}} - y_c^2 \right)} \right) \quad (3.44)$$

$$a = \frac{M_{20}}{M_{00}} - x_c^2 \quad (3.45)$$

$$b = 2 \frac{M_{11}}{M_{00}} - x_c y_c \quad (3.46)$$

$$c = \frac{M_{02}}{M_{00}} - y_c^2 \quad (3.47)$$

$$l = \sqrt{\frac{(a+c) + \sqrt{b^2 + (a-c)^2}}{2}} \quad (3.48)$$

$$w = \sqrt{\frac{(a+c) - \sqrt{b^2 + (a-c)^2}}{2}} \quad (3.49)$$

Rozmiar okna przeszukiwania s jest adaptowany każdorazowo po znalezieniu położenia obiektu, na podstawie zależności:

$$s = 2 \sqrt{\frac{M_{00}}{P_{max}}} \quad (3.50)$$

gdzie:

P_{max} – maksymalna możliwa wartość rozkładu P

W [5] zaproponowano określenie szerokości okna jako $w_s = 2s$ a jego długości jako $l_s = 2,4s$. Wynika to z pierwotnego zastosowania metody, tj. śledzenia twarzy, których kształt jest w przybliżeniu elipsoidalny.

3.3. Filtr Kalmana

Wzbogacić o informacje z [1]

Filtr Kalmana (ang. *Kalman Filter*, w skrócie *KF*) jest rekurencyjnym algorytmem optymalnej estymacji wektora stanu modelu obiektu, zakładającym jego liniowość oraz rozkład normalny amplitudy występującego szumu procesu i pomiarowego (jest rodzajem optymalnego, rekursywnego filtru Bayesa) [6]. Od kiedy został po raz pierwszy opublikowany w roku 1960 przez Rudolfa Kalmana znalazł szerokie zastosowanie, np. w rozwiązaniu [16].

3.3.1. Klasyczny filtr Kalmana

Dany jest liniowy i dyskretny układ dynamiczny opisany równaniami [27]:

$$\mathbf{x}_k = \mathbf{A}\mathbf{x}_{k-1} + \mathbf{B}\mathbf{u}_{k-1} + \mathbf{w}_{k-1} \quad (3.51)$$

$$\mathbf{z}_k = \mathbf{H}\mathbf{x}_k + \mathbf{v}_k \quad (3.52)$$

gdzie:

k – czas

\mathbf{x} – wektor stanu

\mathbf{A} – macierz przejścia, wiążąca bieżący stan z poprzednim

\mathbf{B} – macierz sterowania, wiążąca bieżący stan z sygnałem sterującym

\mathbf{u} – wektor sygnału sterującego

\mathbf{w} – wektor szumu procesu

\mathbf{z} – wektor pomiaru stanu

\mathbf{H} – macierz pomiaru, wiążąca wynik pomiaru ze stanem

\mathbf{v} – wektor szumu pomiaru

Szumy \mathbf{w} i \mathbf{v} są od siebie niezależne, mają postać postać szumu białego, a rozkład prawdopodobieństwa ich amplitud ma postać rozkładu normalnego o zerowej wartości oczekiwanej [27]:

$$p(\mathbf{w}) = N(0, \mathbf{Q}) \quad (3.53)$$

$$p(\mathbf{v}) = N(0, \mathbf{R}) \quad (3.54)$$

gdzie:

\mathbf{Q} – macierz kowariancji szumu procesu

\mathbf{R} – macierz kowariancji szumu pomiarowego

Podsumowując, zakłada się pewną zależność pomiędzy bieżącym a poprzednim stanem obiektu, przy czym może być ona zaburzana w nieznany sposób. Stan obiektu nie jest znany, może być jedynie obserwowany za pośrednictwem pomiarów obarczonych pewnym błędem. Celem KF jest wyznaczenie estymaty stanu tego obiektu, w sposób statystycznie optymalny ze względu na błąd estymacji.

W algorytmie KF rozróżniono estymaty stanu *a priori* $\hat{\mathbf{x}}_k^-$, wyznaczaną na podstawie poprzedniego stanu, oraz *a posteriori* $\hat{\mathbf{x}}_k$, uwzględniającą ostatni wynik pomiarów. Błędy estymacji *a priori* i *a posteriori* definiuje się jako [27]:

$$\mathbf{e}_k^- \equiv \mathbf{x}_k - \hat{\mathbf{x}}_k^- \quad (3.55)$$

$$\mathbf{e}_k \equiv \mathbf{x}_k - \hat{\mathbf{x}}_k \quad (3.56)$$

Analogicznie, macierze kowariancji błędów estymacji *a priori* i *a posteriori* mają postaci:

$$\mathbf{P}_k^- = E[\mathbf{e}_k^- \mathbf{e}_k^{-T}] \quad (3.57)$$

$$\mathbf{P}_k = E[\mathbf{e}_k \mathbf{e}_k^T] \quad (3.58)$$

gdzie:

E – operator wartości oczekiwanej

Elementy macierzy kowariancji błędów leżące na ich głównych przekątnych to wariancje błędów estymacji poszczególnych zmiennych stanu, stanowiące miarę niepewności tej estymacji. Pozostałe elementy to kowariancje pomiędzy błędami estymacji poszczególnych zmiennych stanu, określające ich wzajemne powiązanie. Na podstawie estymaty *a priori* można wyznaczyć estymatę *a posteriori*, zgodnie z zależnością [27]:

$$\hat{\mathbf{x}}_k = \hat{\mathbf{x}}_k^- + \mathbf{K}(\mathbf{z}_k - \mathbf{H}\hat{\mathbf{x}}_k^-) \quad (3.59)$$

gdzie:

K – macierz wzmocnienia Kalmana

Macierz wzmocnienia Kalmana \mathbf{K} powinna być zostać określona tak, aby minimalizować wariancję *a posteriori* \mathbf{P}_k , jedna z jej możliwych postaci może zostać obliczona z równania [27]:

$$\mathbf{K}_k = \mathbf{P}_k^- \mathbf{H}^T (\mathbf{H} \mathbf{P}_k^- \mathbf{H}^T + \mathbf{R})^{-1} \quad (3.60)$$

Z równania 3.60 wynika, że im bardziej elementy macierzy kowariancji błędu pomiarowego \mathbf{H} zbliżają się do zera, tym bardziej wynik równania 3.59 zależy od wyniku pomiaru \mathbf{z}_k i jednocześnie mniej zależy od estymaty *a priori* $\hat{\mathbf{x}}_k^-$. Z drugiej strony, dążenie wartości elementów macierzy kowariancji błędów estymacji *a priori* do zera implikuje spadek znaczenia pomiaru \mathbf{z}_k i wzrost znaczenia przewidywanego wyniku $\hat{\mathbf{x}}_k^-$ [27].

Algorytm estymacji stanu z wykorzystaniem *KF* składa się dwóch etapów. Pierwszy z nich, określany jako faza predykcji, polega na wyznaczeniu nowej estymaty stanu oraz macierzy kowariancji błędów *a priori* na podstawie ich poprzednich postaci [27]:

$$\hat{\mathbf{x}}_k^- = \mathbf{A}\hat{\mathbf{x}}_{k-1} + \mathbf{B}\mathbf{u}_{k-1} \quad (3.61)$$

$$\mathbf{P}_k^- = \mathbf{A}\mathbf{P}_{k-1}\mathbf{A}^T + \mathbf{Q} \quad (3.62)$$

W pierwszy kroku zakłada się początkowe wartości $\hat{\mathbf{x}}_{k-1}$ oraz \mathbf{P}_{k-1} . Po fazie predykcji następuje faza korekcji, w której uwzględniając wyniki pomiarów wyznacza się wartości *a posteriori* [27]:

$$\mathbf{K}_k = \mathbf{P}_k^- \mathbf{H}^T (\mathbf{H} \mathbf{P}_k^- \mathbf{H}^T + \mathbf{R})^{-1} \quad (3.63)$$

$$\hat{\mathbf{x}}_k = \hat{\mathbf{x}}_k^- + \mathbf{K}_k(\mathbf{z}_k - \mathbf{H}\hat{\mathbf{x}}_k^-) \quad (3.64)$$

$$\mathbf{P}_k = (\mathbf{I} - \mathbf{K}_k \mathbf{H})\mathbf{P}_k^- \quad (3.65)$$

Po zakończeniu iteracji wartości *a posteriori* traktowane są jako wartości *a priori* dla kolejnego kroku $k+1$. Macierz kowariancji szumu pomiarowego \mathbf{R} można zwykle wyznaczyć eksperymentalnie. Bardziej kłopotliwe jest określenie postaci macierzy kowariancji szumu procesu \mathbf{Q} . Przy założeniu dużej dokładności pomiarów przyjęcie

dużych wartości elementów \mathbf{Q} , a więc dużej niepewności procesu, może przynieść zadowalające rezultaty. Innym rozwiązaniem jest wykonanie identyfikacji systemu poprzez dobór wartości macierzy \mathbf{R} i \mathbf{Q} z wykorzystaniem np. drugiego filtru Kalmana [27].

3.3.2. Rozszerzony filtr Kalmana

Jak wspomniano wcześniej, zastosowanie KF w klasycznej postaci ogranicza się do układów liniowych. W przypadku układów nieliniowych (opisanych równaniami 3.66 i 3.67) znajduje zastosowanie jego zmodyfikowana wersja, nosząca nazwę **rozszerzonego filtru Kalmana** (ang. *Extended Kalman Filter*, w skrócie *EKF*) [27].

$$\mathbf{x}_k = f(\mathbf{x}_{k-1}, \mathbf{u}_{k-1}, \mathbf{w}_{k-1}) \quad (3.66)$$

$$\mathbf{z}_k = h(\mathbf{x}_k, \mathbf{v}_k) \quad (3.67)$$

gdzie:

f – nieliniowa funkcja wiążąca bieżący stan ze stanem poprzednim, sygnałem sterowania oraz szumem procesu
 h – nieliniowa funkcja wiążąca wynik pomiaru ze stanem oraz szumem pomiarowym

Ponieważ konkretne wartości szumów \mathbf{w}_k i \mathbf{v}_k nie są znane, wektor stanu oraz wektor wyników pomiarów można aproksymować przez ich pominięcie [27]:

$$\tilde{\mathbf{x}}_k = f(\hat{\mathbf{x}}_{k-1}, \mathbf{u}_{k-1}, 0) \quad (3.68)$$

$$\tilde{\mathbf{z}}_k = h(\tilde{x}_k, 0) \quad (3.69)$$

gdzie:

$\tilde{\mathbf{x}}_k$ – aproksymacja wektora stanu w chwili k

$\tilde{\mathbf{z}}_k$ – aproksymacja wyników pomiarów w chwili k

$\hat{\mathbf{x}}_k$ – estymata stanu *a posteriori* z poprzedniego kroku k

Istotną wadą *EKF* jest brak zachowania normalnego rozkładu szumów po poddaniu ich nieliniowym przekształceniom [27]. W celu dokonania estymacji wykonywana jest linearyzacja równań 3.67 i 3.66 poprzez ich rozwinięcie w szereg Taylora w otoczeniu przybliżeń 3.68 i 3.69 [27]:

$$\mathbf{x}_k \approx \tilde{\mathbf{x}}_k + \mathbf{A}_k(\mathbf{x}_{k-1} - \hat{\mathbf{x}}_{k-1}) + \mathbf{W}_k \mathbf{w}_{k-1} \quad (3.70)$$

$$\mathbf{z}_k \approx \tilde{\mathbf{z}}_k + \mathbf{H}_k(\mathbf{x}_k - \tilde{x}_k) + \mathbf{V}_k \mathbf{v}_k \quad (3.71)$$

gdzie:

\mathbf{A}_k – macierz Jacobiego pochodnych cząstkowych funkcji f ze względu na \mathbf{x} w chwili k

\mathbf{W}_k – macierz Jacobiego pochodnych cząstkowych funkcji f ze względu na \mathbf{w} w chwili k

\mathbf{H}_k – macierz Jacobiego pochodnych cząstkowych funkcji h ze względu na \mathbf{x} w chwili k

\mathbf{V}_k – macierz Jacobiego pochodnych cząstkowych funkcji h ze względu na \mathbf{v} w chwili k

Elementy macierzy \mathbf{A}_k , \mathbf{W}_k , \mathbf{H}_k i \mathbf{V}_k zdefiniowane są następująco:

$$\mathbf{A}_{k,[i,j]} = \frac{\partial f_{[i]}}{\partial \mathbf{x}_{[j]}}(\hat{\mathbf{x}}_{k-1}, \mathbf{u}_{k-1}, 0) \quad (3.72)$$

$$\mathbf{W}_{k,[i,j]} = \frac{\partial f_{[i]}}{\partial \mathbf{w}_{[j]}}(\hat{\mathbf{x}}_{k-1}, \mathbf{u}_{k-1}, 0) \quad (3.73)$$

$$\mathbf{H}_{k,[i,j]} = \frac{\partial h_{[i]}}{\partial \mathbf{x}_{[j]}}(\tilde{\mathbf{x}}_k, 0) \quad (3.74)$$

$$\mathbf{V}_{k,[i,j]} = \frac{\partial h_{[i]}}{\partial \mathbf{v}_{[j]}}(\tilde{\mathbf{x}}_k, 0) \quad (3.75)$$

Zakłada się definicję błędów przybliżeń wektora stanu $\tilde{\mathbf{e}}_{\mathbf{x}_k}$ i pomiarów $\tilde{\mathbf{e}}_{\mathbf{z}_k}$:

$$\tilde{\mathbf{e}}_{\mathbf{x}_k} \equiv \mathbf{x}_k - \tilde{\mathbf{x}}_k \quad (3.76)$$

$$\tilde{\mathbf{e}}_{\mathbf{z}_k} \equiv \mathbf{z}_k - \tilde{\mathbf{z}}_k \quad (3.77)$$

Po wstawieniu równań 3.76 i 3.77 do równań 3.70 i 3.71 przekształceniach [27]:

$$\tilde{\mathbf{e}}_{\mathbf{x}_k} \approx \mathbf{A}_k(\mathbf{x}_{k-1} - \hat{\mathbf{x}}_{k-1}) + \epsilon_k \quad (3.78)$$

$$\tilde{\mathbf{e}}_{\mathbf{z}_k} \approx \mathbf{H}_k \tilde{\mathbf{e}}_{\mathbf{x}_k} + \eta_k \quad (3.79)$$

gdzie:

ϵ_k – niezależna zmienna losowa o zerowej wartości średniej i macierzy kowariancji $\mathbf{WQ}_k \mathbf{W}^T$

η_k – niezależna zmienna losowa o zerowej wartości średniej i macierzy kowariancji $\mathbf{VR}_k \mathbf{V}^T$

Równania 3.78 i 3.79 są liniowe i mają podobną postać jak równania procesu 3.51 i pomiaru 3.52 *KF*, co sugeruje wykorzystanie drugiego, hipotetycznego filtru Kalmana do estymacji błędu predykcji $\tilde{\mathbf{e}}_{\mathbf{x}_k}$ [27]. Jego estymacja, oznaczona jako $\hat{\mathbf{e}}_k$, może następnie służyć do wyznaczenia estymaty stanu *a posteriori* układu nieliniowego, na podstawie równania 3.76:

$$\hat{\mathbf{x}}_k = \tilde{\mathbf{x}}_k + \hat{\mathbf{e}}_k \quad (3.80)$$

Zmienne losowe ϵ_k i η_k mają w przybliżeniu rozkłady prawdopodobieństwa odpowiednio $p(\epsilon_k) \sim N(0, \mathbf{WQ}_k \mathbf{W}^T)$ i $p(\eta_k) \sim N(0, \mathbf{VR}_k \mathbf{V}^T)$. Korzystając z równania 3.59, oraz zakładając zerową wartość predykcji $\hat{\mathbf{e}}_k$ [27]:

$$\hat{\mathbf{e}}_k = \mathbf{K}_k \tilde{\mathbf{e}}_{\mathbf{z}_k} \quad (3.81)$$

Wstawienie równania 3.81 do 3.80 oraz podstawienie aproksymowanej wartości błędu pomiaru na postawie równania 3.77 ujawnia brak konieczności stosowania drugiego *KF* i pozwala na wyznaczenie postaci równania korekcji stanu *EKF* [27]:

$$\hat{\mathbf{x}}_k = \tilde{\mathbf{x}}_k + \mathbf{K}_k(\mathbf{z}_k - \tilde{\mathbf{z}}_k) \quad (3.82)$$

Macierz wzmacnienia Kalmana \mathbf{K}_k wyznacza się analogicznie jak w *KF*, na podstawie równania 3.65, podając zmodyfikowaną postać macierzy kowariancji błędów pomiaru.

Równania *EKF* fazy predykcji przyjmują następującą postać [27]:

$$\hat{\mathbf{x}}_k^- = f(\mathbf{x}_{k-1}, \mathbf{u}_{k-1}, 0) \quad (3.83)$$

$$\mathbf{P}_k^- = \mathbf{A}_k \mathbf{P}_{k-1} \mathbf{A}_k^T + \mathbf{W}_k \mathbf{Q}_{k-1} \mathbf{W}_k^T \quad (3.84)$$

Równania fazy korekcji [27]:

$$\mathbf{K}_k = \mathbf{P}_k^- \mathbf{H}_k^T (\mathbf{H}_k \mathbf{P}_k^- \mathbf{H}_k^T + \mathbf{V}_k \mathbf{R}_k \mathbf{V}_k^T)^{-1} \quad (3.85)$$

$$\hat{\mathbf{x}}_k = \hat{\mathbf{x}}_k^- + \mathbf{K}_k (\mathbf{z}_k - h(\hat{\mathbf{x}}_k^-, 0)) \quad (3.86)$$

$$\mathbf{P}_k = (\mathbf{I} - \mathbf{K}_k \mathbf{H}_k) \mathbf{P}_k^- \quad (3.87)$$

3.3.3. Filtr Kalmana w śledzeniu wizyjnym

Zastosowanie *KF* wymaga założenia pewnego modelu obiektu, którego stan jest estymowany. W przypadku wizyjnego śledzenia obiektu, zmienne stanu modelu (lub przynajmniej ich część) będą opisywały parametry kinematyczne tego obiektu, tzn. jego położenie, prędkość i przyspieszenie. Poniżej przedstawiono cztery warianty ogólnego liniowego modelu ruchu obiektu mogące znaleźć zastosowanie w śledzeniu z wykorzystaniem *KF* według [11].

Model **dryfujących punktów** (ang. *drifting points*) ma najprostszą postać i zakłada jednostkową postać macierzy przejścia $\mathbf{A} = \mathbf{I}$ przy wektorze stanu przechowującym położenie obiektu \mathbf{p} , tzn. w zasadzie obiekt stacjonarny. Zmiana stanu odbywa się wyłącznie poprzez występujący szum procesu. Model tej postaci znajduje zastosowanie w przypadku braku możliwości zdefiniowania dokładniejszych założeń odnośnie dynamiki obiektu [11].

W modelu **stałoprędkościowym** (ang. *constant velocity*) zakłada się, że wektor stanu przechowuje informacje o położeniu \mathbf{p} i prędkości \mathbf{v} obiektu, przy czym prędkość nie ulega zmianie, stąd [11]:

$$\mathbf{x} = \begin{bmatrix} \mathbf{p} \\ \mathbf{v} \end{bmatrix} \quad (3.88)$$

$$\mathbf{A} = \begin{bmatrix} \mathbf{I} & (\Delta t) \mathbf{I} \\ 0 & \mathbf{I} \end{bmatrix} \quad (3.89)$$

gdzie:

Δt – okres próbkowania

Analogicznie, model **stałoprzyspieszeniowy** rozszerza wektor stanu o przyspieszenie \mathbf{a} , oraz zakłada jego niezmienność. Wówczas wektor stanu oraz macierz przejścia przyjmują postaci [11]:

$$\mathbf{x} = \begin{bmatrix} \mathbf{p} \\ \mathbf{v} \\ \mathbf{a} \end{bmatrix} \quad (3.90)$$

$$\mathbf{A} = \begin{bmatrix} \mathbf{I} & (\Delta t)\mathbf{I}0 \\ 0 & \mathbf{I} & (\Delta t)\mathbf{I} \\ 0 & 0 & \mathbf{I} \end{bmatrix} \quad (3.91)$$

Ostatni z ogólnych modeli zakłada **ruch okresowy**, w którym położenie p spełnia zależność $\frac{d^2 p}{dt^2} = -p$. Przy założeniu, że wektor stanu przechowuje położenie p oraz prędkość v , macierz przejścia ma postać [11]:

$$\mathbf{x} = \begin{bmatrix} p \\ v \end{bmatrix} \quad (3.92)$$

$$\mathbf{A} = \begin{bmatrix} 1 & \Delta t \\ -\Delta t & 1 \end{bmatrix} \quad (3.93)$$

W wizyjnym śledzeniu obiektów funkcją *KF* jest ograniczenie przestrzeni poszukiwań obiektu na nowej klatce sekwencji video (patrz 1.2.3), co praktycznie jest realizowane poprzez przyjęcie pewnego okna, w którym znajduje się obiekt oraz estymacji jego kolejnej pozycji [23]. Faza pomiaru może być realizowana, np. poprzez próbę wykrycia obiektu wewnętrz nowego okna [11].

4. Przegląd istniejących rozwiązań

Więcej przykładów

4.1. Moduł sterowania kamerą *pan-tilt* dla pojazdu *UAV*

W publikacji [21] przedstawiono system sterowania głowicą *pan-tilt* z kamerą wizyjną, zamontowaną na pojazdzie *UAV* (rysunek 4.1). Jak wskazują autorzy, w przypadku aplikacji tego typu, w kontekście śledzenia wizyjnej znacznym problemem jest połączenie zjawiska wibracji całego pojazdu i długiej ogniskowej kamery. Zastosowanie platformy *pan-tilt* ułatwia śledzenie i wykrawanie obiektów zainteresowanie, nie wymaga bowiem zmiany orientacji całego pojazdu.



Rysunek 4.1: Bezzałogowy helikopter - platforma testowa

Źródło: [21]

Obiekt zainteresowania jest wykrywany przez system poprzez wykrawanie punktów zainteresowania według algorytmu Harrisa, w którym narzucono dodatkowo kryterium maksymalnej wariancji przy zwiększaniu okna wykrawania oraz minimalnego progu modułu wartości własnych. Pozwoliło to na wybranie punktów zainteresowania o największej odporności oraz wykorzystanie ich jako reprezentacji śledzonego obiektu w algorytmie Lucasa-Kanade (wersja klasyczna, opisana w rozdziale 3.1.1). Zwracany przez niego błąd śledzenia (rozbieżność pomiędzy położeniem obiektu a środkiem obrazu) wraz z różnicą pomiędzy jego bieżącą o poprzednią wartością stanowi sprzężenie zwrotne dla dwóch kontrolerów działających w oparciu o wnioskowanie rozmyte ang. *fuzzy*

controllers), które sterują pracą silników obydwu osi platformy *pan-tilt*. Wyjście kontrolerów stanowi względne przemieszczenie kątowe, które musi zostać wykonane w celu wycentrowania obiektu w polu widzenia kamery i jest wyznaczane na podstawie 49 zasad wnioskowania.

Działanie systemu zostało przetestowane przy wykorzystaniu pojedynczej kamery video o rozdzielczości 320×240 pikseli, środowiskiem działania był system Linux, zainstalowany na pokładowym komputerze VIA mini-ITX (częstotliwość taktowania 1,5GHz, 1Gb pamięci RAM).

4.2. Hybrydowy algorytm śledzenia i podążania za ludźmi dla robota kroczącego

Celem autorów publikacji [17] było opracowanie algorytmu śledzenia i podążania za ludźmi dla robota kroczącego, który umożliwiałby stały nadzór osób starszych, pacjentów szpitali, etc. Jako platformę testową wykorzystano robota Sony AIBO, wskazując na jego zalety w postaci wyposażenia w szereg sensorów (w tym kamerę video) oraz nierzucający się w oczy wygląd. Istotnym wymaganiem wobec opracowanego rozwiązania było osiągnięcie odporności na nieprzewidywalny i nieregularny ruch kamery, wynikający ze sposobu poruszania się robota, oraz zmian w oświetleniu obiektu zainteresowania.

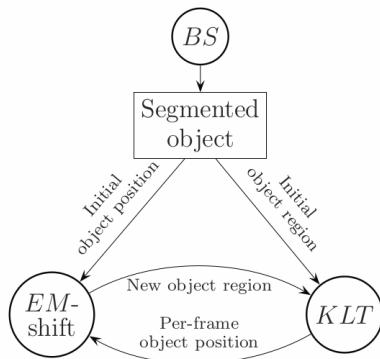
Uproszczony przebieg algorytmu wizyjnego śledzenia obiektu przedstawiono na rysunku 4.2. Wykrycie obiektu następuje poprzez segmentację tła i pierwszego planu (obiektów poruszających się) z wykorzystaniem metody *Gaussian Mixture Model*. Następnie obiekt jest śledzony z wykorzystaniem algorytmu *EM-Shift*, stanowiącego rozwinięcie metody *Mean Shift* (rozdział 3.2.1), w którym dodatkowo wykonywana jest estymacja i adaptacja okna przeszukiwania. Obiekt jest również śledzony z wykorzystaniem algorytmu Lucas-Kanade-Tomasi (rozdział 3.1.2) i wyznaczonych według niego punktów zainteresowania. Krok początkowy (segmentacja poruszających się obiektów) nie może realizować bezpośrednio zadania śledzenia ze względu na możliwy ruch kamery. Algorytm *EM-Shift* działa w oparciu o histogram obiektu zainteresowania, co w wypadku zmiany oświetlenia może prowadzić do zgubienia obiektu. Z kolei *KLT* nie dokonuje segmentacji obiektu jako takiej, przez co nie można stwierdzić czy śledzone przez niego punkty faktycznie należą do obiektu. Połączenie tych dwóch metod śledzenia obiektów pozwala na kompensację ich wad i uzyskanie odpornego rozwiązania.

Pokładowa kamera robota ma rozdzielczość 208×160 pikseli, obliczenia w trakcie testów były wykonywane zdalnie, za pośrednictwem sieci WiFi, na komputerze PC z procesorem AMD 3500+ i 2Gb pamięci RAM.

4.3. Algorytm wykrywania, śledzenia i podążania za obiektem na obrazie z kamery sferycznej dla robota mobilnego

W pracy [20] zaprezentowano algorytm wykrywania, śledzenia i podążania za obiektem, którego czujnik wizyjny stanowi kamera sferyczna (ang. *omnidirectional camera*), dostarczająca kompletnego obrazu sceny w promieniu 360° (rysunek 4.3). Przy zastosowaniu kamery tego typu nie istnieje możliwość ucieczki obiektu zainteresowania poza pole widzenia, w związku z tym system sterowania nie musi realizować funkcji utrzymywania go w centrum obrazu. Zakłada się jednoczesny ruch zarówno obiektu zainteresowania jak i robota. Jako platformę testową wykorzystano robota kołowego Pioneer 3DX oraz kamerę z obiektywem typu *fisheye*.

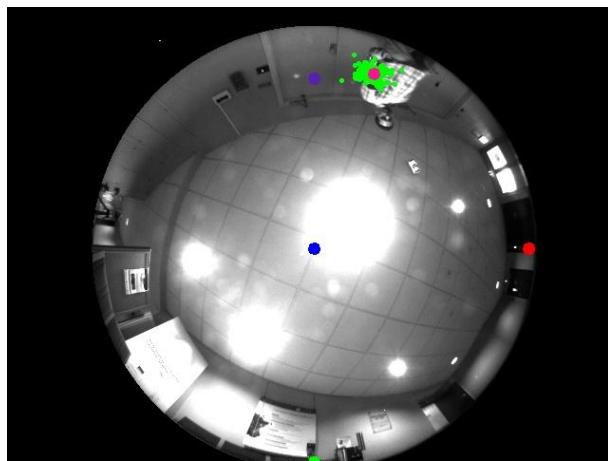
Ze względu na rodzaj stosowanego obiektywu konieczne jest przeprowadzenie kalibracji kamery. Polega ona na wyznaczeniu jej wewnętrznego modelu na podstawie którego rejestrowany obraz jest przekształcany w celu



Rysunek 4.2: Przebieg algorytmu wizyjnego śledzenia obiektu

Od góry, od lewej: odejmowanie tła - segmentacja obiektu; algorytm *EM-shift* - wykrywanie nowego obszaru zainteresowania; algorytm Lucasa-Kanade-Tomasi - wykrywanie nowych współrzędnych obiektu

Źródło: [17]



Rysunek 4.3: Przykładowy obraz z kamery sferycznej

Źródło: [20]

usunięcia zniekształceń wnoszonych przez układ optyczny. W pierwszym kroku algorytm wyznacza rzadki przepływ optyczny z wykorzystaniem algorytmu Lucasa-Kanade w wersji piramidowej (rozdział 3.1.3). Następnie wektory przepływu optycznego są rozkładane na składową wynikającą z ruchu robota, wyznaczoną na podstawie odometrii, oraz ewentualną składową wynikającą z ruchu obecnego, poruszającego się obiektu. Zastosowanie kamery sferycznej komplikuje znacznie tą operację, ponieważ manipulowanie punktami obrazu musi odbywać się w przestrzeni współrzędnych sferycznych. Składowe wektorów przepływu optycznego sklasyfikowane jako wynik ruchu obiektów na obrazie są następnie dzielone na grupy, dla których wyznacza się reprezentację pojedynczym wektorem w przestrzeni sferycznej, traktowanym jako wynik pomiarów.

W testowanym rozwiązaniu w przypadku wykrycia wielu obiektów wybierany i śledzony był jedynie najbliższy. Na podstawie wyników pomiarów konstruowany jest probabilistyczny model w postaci rozkładu prawdopodobieństwa von Misesa-Fishera. Właściwe zadanie śledzenia realizowane jest poprzez estymację stanu obiektu z

wykorzystaniem rekurencyjnego estymatora Bayesa (którego szczególnym przypadkiem, zakładającym liniowość modelu i rozkłady normalne zmiennych losowych, jest filtr Kalmana opisany w rozdziale 3.3.1). Finalnie, algorytm śledzenia zwraca estymowany kierunek ruchu obiektu. Na jego podstawie system sterowania wykonuje dodatkowo zadanie utrzymania obiektu w konkretnym, uprzednio zdefiniowanym, obszarze obrazu z kamery. W tym celu estymata kierunku ruchu obiektu jest rzutowana na płaszczyznę obrazu (do współrzędnych sferycznych) i stosowane jest odpowiednie prawo sterowania.

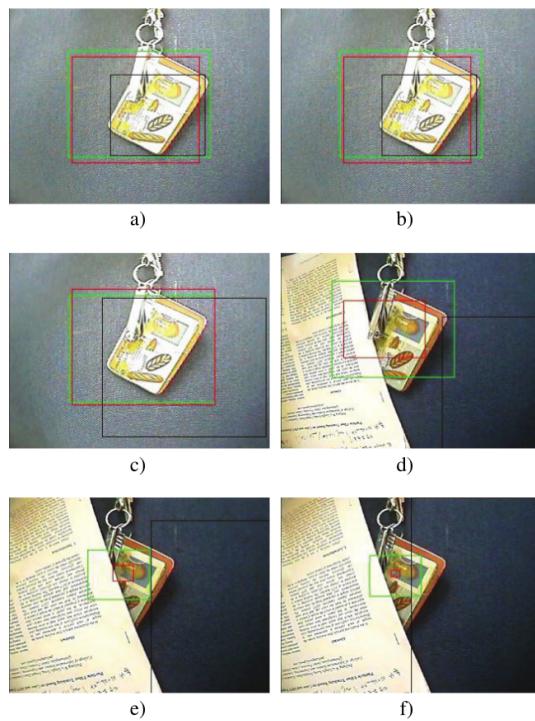
4.4. System wizyjnego śledzenia obiektów w czasie rzeczywistym do zastosowania w wizyjnym sterowaniu robota mobilnego

W [29] opisany został algorytm śledzenia wizyjnego przeznaczony do zastosowania w systemie sterowania robota mobilnego. Jako założenie projektowe przyjęto odporność na typowe problemy spotykane powiązane ze środowiskiem jego działania (szybki ruch obiektu, przysłonięcia, deformacje, zmiany oświetlenia, mała odrębność tła i obiektu) oraz zdolność wykonania w czasie rzeczywistym. Zwracane przez algorytm położenie obiektu jest przekazywane do systemu sterowania robota. Jako platformę testową wykorzystano robota HBE-RoboCAR-Embedded i jego kamerę pokładową.

Autorzy publikacji przedstawiają propozycję algorytmu, nazwanego *Modified CamShift Guided Particle Filter* (w skrócie *MCAMSGPF*), integrującego filtr cząsteczkowy z algorymem *CAMSHIFT* (opisanym w rozdziale 3.2.2). Algorytm jest w istocie rozwinięciem koncepcji podobnego rozwiązania o nazwie *CAMSHIFT Guided Particle Filter*, które jednak nie było implementowane w robotyce mobilnej i dlatego nie zostało tutaj przytoczone. Filtr cząsteczkowy (określany inaczej jako sekwencyjna metoda Monte Carlo) jest metodą estymacji stanu, w której rozkład prawdopodobieństwa zmiennych stanu reprezentuje się poprzez zbiór próbek (cząsteczek) z przypisanymi wagami. Każda z cząsteczek reprezentuje możliwy punkt trajektorii stanu, rozkład prawdopodobieństwa jest propagowany w czasie z wykorzystaniem rekursywnego filtra Bayesa, estymacja wartości zmiennych stanu odbywa się poprzez znalezienie w każdym kroku cząsteczki o najwyższej wadze [8]. Procedura *MCAMSGPF* rozpoczyna się od propagacji poprzedniego, znanego stanu obiektu.

Na podstawie założonego dynamicznego modelu w postaci procesu autoregresywnego wyznaczane jest nowe położenie obiektu, które następnie przesuwane jest do minimum lokalnego funkcji gęstości prawdopodobieństwa estymowanej z wykorzystaniem histogramu przestrzeni kolorów *HSV* obrazu. W końcowej fazie propagacji, położenie obiektu (czyli pojedyncza cząsteczka) jest przesuwana na podstawie rozkładu prawdopodobieństwa stanowiącego połączenie rozkładu normalnego z założonym modelem dynamicznym. Na podstawie otrzymanego stanu wykonywany jest model obserwacyjny, oparty o punkty charakterystyczne *SURF* (*Speeded Up Robust Features*). Metoda *SURF* jest modyfikacją metody *SIFT* przytoczonej w rozdziale 2.1.1, której celem było przyspieszenie procesu ekstrakcji i deskrypcji. Punkty *SURF* są invariantne, co zapewnia odporność całego modelu. Model obserwacyjny służy następnie do określenia wag cząsteczek. Cała przedstawiona wyżej procedura jest wykonywana dla każdej cząsteczki. Ostatnim krokiem algorytmu jest odrzucenie cząsteczek o zbyt niskich wagach.

Połączenie filtra cząsteczkowego z algorytmem *CAMSHIFT* i modelem opartym o punkty charakterystyczne *SURF* pozwoliło na uzyskanie wysokiej odporności na błędny śledzenia wynikające z przysłonięć obiektu oraz podobieństwa kolorów pikseli tła i obiektu, co jest słabą stroną samej metody *CAMSHIFT* (przykład przedstawiono na rysunku 4.4).



Rysunek 4.4: Porównanie działania algorytmu *MCAMSGPF* z innymi metodami

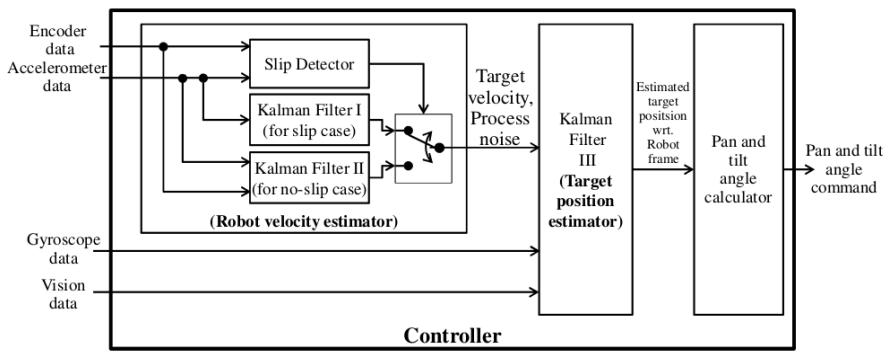
Na obrazach a) - f) przedstawiono klatki wybrane klatki sekwencji video, w której nastąpiło przysłonięcie obiektu zainteresowania. Objasnienie oznaczeń: zielony prostokąt - obiekt śledzony algorytmem *MCAMSGPF*; czerwony prostokąt - obiekt śledzony algorytmem *CAMSGPF*; obiekt śledzony algorytmem *CAMSHIFT*

Źródło: [29]

4.5. System śledzenia obiektów z wykorzystaniem kamery *pan-tilt* zamontowanej na robocie mobilnym

W artykule [16] zaprezentowano system sterowania głowicą *pan-tilt* z zamontowaną kamerą wizyjną, przeznaczony do zastosowania w robotyce mobilnej, którego funkcją jest utrzymanie obiektu zainteresowania w polu widzenia kamery. W ramach tego zadania estymowane są trójwymiarowe współrzędne obiektu zainteresowania w odniesieniu do lokalnego układu współrzędnych robota oraz obliczane są odpowiednie przemieszczenia kątowe dla silników orientujących kamerę. Zakładany jest ruch robota w przestrzeni trójwymiarowej przy nieruchomym obiekcie zainteresowania.

W przeciwieństwie do innych przykładów rozwiązań przytaczanych w niniejszej pracy, tematyka publikacji [16] abstrahuje od algorytmów przetwarzania obrazów, skupiając się wyłącznie na systemie sterowania. Zadanie śledzenia obiektu realizowane jest w oparciu o fuzję danych pomiarowych oraz estymację położenia obiektu. Robot stanowiący platformę testową wyposażony jest w enkodery kół, trójosiowy akcelerometr, trójosiowy żyroskop, enkodery silników mechanizmu *pan-tilt* oraz kamerę wizyjną o rozdzielcości 640×480 pikseli. Do celów testowych, jako obiekt zainteresowania wykorzystywany jest fotografia twarzy, natomiast pomiar polega na wykonaniu odpowiedniego algorytmu wykrywania. System sterowania, którego schemat przedstawia rysunek 4.5, składa się z trzech zasadniczych modułów - estymatora prędkości robota, estymatora położenia obiektu zainteresowania i



Rysunek 4.5: Schemat systemu sterowania głowicą *pan-tilt*

Źródło: [16]

kalkulatora współrzędnych złączowych mechanizmu *pan-tilt*.

Dane wejściowe dla estymatora prędkości stanowią pomiary z enkoderów kół robota oraz akcelerometru. Pomiary wykonywane przy użyciu enkoderów są dokładne, o ile nie występuje utrata przyczepności kół robota (ang. *slip*), w związku z tym są one porównywane ze wskazaniami akcelerometru i w przypadku zbyt dużej rozbieżności stwierdzane jest wystąpienie tego zjawiska. Następnie wybierany jest jeden z dwóch estymatorów prędkości, z których oba mają postać klasycznego filtra Kalmana (opisanego w rozdziale 3.3.1). Pierwszy z nich, przeznaczony do zastosowania w przypadku wystąpienia utraty przyczepności, estymuje prędkość wyłącznie na podstawie wyników pomiaru akcelerometrem. Drugi, używany przy zachowaniu przyczepności, uwzględnia również pomiary wykonane enkoderami kół, dzięki czemu zapewnia większą dokładność. Moduł estymatora prędkości zwraca pożorną (wynikającą z ruchu robota) prędkość ruchu obiektu zainteresowania w lokalnym układzie współrzędnych robota oraz szum procesu.

Estymator położenia obiektu zainteresowania ma postać rozszerzonego filtra Kalmana (opisanego w rozdziale 3.3.2). Poza danymi zwracanymi przez estymator prędkości, wykorzystuje on pomiar wizyjny (położenie obiektu na obrazie z kamery) oraz obecną orientację kamery mierzoną przez żyroskop. Dane wyjściowe stanowią estymowane położenia obiektu zainteresowania w układzie współrzędnych robota, na podstawie który moduł kalkulatora wyznacza wartości współrzędnych złączowych, pozwalające na utrzymanie obiektu zainteresowania w polu widzenia. Częstotliwość próbkowania kamery (27Hz) jest niższa, niż pozostałych sensorów (108Hz). Z tego powodu, pomiędzy kolejnymi klatkami ostatni wyznaczony stan jest propagowany.

4.6. Podsumowanie

Rozszerzyć, krótkie omówienie podejść i algorytmów, mocne/słabe strony, porównanie efektywności algorytmów

Wśród algorytmów stosowanych w wizyjnym śledzeniu obiektów wyróżnić można dwie grupy:

1. Algorytmy tworzące pewną reprezentację obiektu na obrazie i poszukujące jego najbardziej prawdopodobnej instancji na kolejnych klatkach sekwencji video, do których należy algorytm Lucasa-Kanade (3.1) oraz algorytm Mean-Shift (3.2).

2. Algorytmy dokonujące predykcji trajektorii obiektu na obrazie, do których należą metody oparte o rekurencyjny filtr Bayesa, tj. filtr Kalmana (3.3) i filtr cząsteczkowy

Pomimo pozornie różnych funkcji, obie grupy mogą stanowić źródło sprzężenia zwrotnego dla systemu sterowania robota mobilnego. Zaznaczyć należy, że zastosowanie algorytmów z grupy drugiej wymaga wdrożenia dodatkowego mechanizmu pomiaru położenia obiektu, np. poprzez okresowe wykonanie procedury wykrywania go na podstawie wzorca.

Obszar aplikacji w postaci robotyki mobilnej dodatkowo utrudnia nietrywialne z natury zadanie śledzenia obiektów. Poza ruchem samego obiektu należy brać pod uwagę możliwość ruchu samego robota, skutującą zwiększeniem częstości występowania niepożądanych zjawisk, takich jak przysłonięcia obiektu zainteresowania, zmiany oświetlenia sceny czy niewielka rozbieżność pomiędzy wyglądem obiektu zainteresowania a tłem. Dodatkowo, ruch kamery zmienia wynik działania niektórych powszechnie stosowanych metod, np. w polu przepływu optycznego pojawia się składowa stała, reprezentująca pozorny ruch całej sceny. W przypadku platformy sprzętowej w postaci robota mobilnego często należy liczyć się z jej niewielką wydajnością obliczeniową.

W celu zwiększenia ogólnej odporności systemu śledzenia, częstym rozwiązaniem jest wykorzystanie więcej niż jednego algorytmu śledzenia wizyjnego ([17], [29], [20]). W takich przypadkach dane zwracane przez pierwszy stanowią dane wejściowe kolejnego, co pozwala na wzajemną kompensację nieprawidłowego działania.

Standardowym typem stosowanego czujnika wizyjnego jest pojedyncza kamera video o niskiej rozdzielczości ([17], [21], [29], [15], wyjątek stanowi [20]). Jest to zrozumiałe, ze względu na wspomniane wcześniej ograniczenie mocy obliczeniowej, istotne w przypadku komputerów pokładowych.

Bibliografia

- [1] M. Sanjeev Arulampalam, Simon Maskell, Neil Gordon i Tim Clapp. „A tutorial on particle filters for online nonlinear/non-Gaussian Bayesian tracking”. W: *IEEE Transactions on Signal Processing* 50.2 (lut. 2002), s. 174–188. DOI: [10.1109/78.978374](https://doi.org/10.1109/78.978374).
- [2] Simon Baker i Iain Matthews. „Lucas-Kanade 20 Years On: A Unifying Framework”. W: *International Journal of Computer Vision* 56.3 (lut. 2004), s. 221–255. DOI: [10.1023/B:VISI.0000011205.11775.fd](https://doi.org/10.1023/B:VISI.0000011205.11775.fd).
- [3] Simon Baker, Daniel Scharstein, J. P. Lewis, Stefan Roth, Michael J. Black i Richard Szeliski. „A Database and Evaluation Methodology for Optical Flow”. W: *International Journal of Computer Vision* 92.1 (mar. 2011), s. 1–31. DOI: [10.1007/s11263-010-0390-2](https://doi.org/10.1007/s11263-010-0390-2).
- [4] Jean-Yves Bouguet. *Pyramidal implementation of the Lucas Kanade feature tracker - Description of the algorithm*. Spraw. tech. Intel Corporation - Microprocessor Research Labs, 2000.
- [5] Gary R. Bradski. „Real time face and object tracking as a component of a perceptual user interface”. W: *Fourth IEEE Workshop Applications of Computer Vision (WACV)*. USA, NJ, Princeton: IEEE, paź. 1998, s. 214–219. DOI: [10.1109/ACV.1998.732882](https://doi.org/10.1109/ACV.1998.732882).
- [6] Subhash Challa, Mark R. Morelande, Darko Mušicki i Robin J. Evans. *Fundamentals of Object Tracking*. Cambridge University Press, 2011. DOI: <http://dx.doi.org/10.1017/CBO9780511975837>.
- [7] Francois Chaumette i Seth Hutchinson. „Visual servo control. I. Basic approaches”. W: *IEEE Robotics & Automation Magazine* 13.4 (grud. 2006), s. 82–90. DOI: [10.1109/MRA.2006.250573](https://doi.org/10.1109/MRA.2006.250573).
- [8] Yu-Cheng Chou i Bo-Shiun Huang. „Mono vision particle filter based object tracking with a mobile robot”. W: *IEEE/ASME International Conference on Mechatronics and Embedded Systems and Applications (MESA)*. China, Suzhou: IEEE, lip. 2012, s. 87–92. DOI: [10.1109/MESA.2012.6275542](https://doi.org/10.1109/MESA.2012.6275542).
- [9] Dorin Comaniciu i Peter Meer. „Mean shift analysis and applications”. W: *The Proceedings of the Seventh IEEE International Conference on Computer Vision*. T. 2. Greece, Kerkyra: IEEE, wrz. 1999, s. 1197–1203. DOI: [10.1109/ICCV.1999.790416](https://doi.org/10.1109/ICCV.1999.790416).
- [10] Dorin Comaniciu, Visvanathan Ramesh i Peter Meer. „Kernel-based object tracking”. W: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 25.5 (maj 2003), s. 564–577. DOI: [10.1109/TPAMI.2003.1195991](https://doi.org/10.1109/TPAMI.2003.1195991).
- [11] David A. Forsyth i Jean Ponce. *Computer Vision - A modern approach*. 2 wydr. Prentice Hall, 2012.
- [12] Berthold K. P. Horn i Brian G. Schunck. „Determining Optical Flow”. W: *Artificial Intelligence* 17 (1981), s. 185–203.
- [13] Bernd Jähne. *Digital Image Processing*. 6 wydr. Springer-Verlag Berlin Heidelberg, 2005.

- [14] Bahadir Karasulu i Serdar Korukoglu. *Performance Evaluation Software - Moving Object Detection and Tracking in Video*. 1 wydr. SpringerBriefs in Computer Science. Springer-Verlag New York, 2013. DOI: [10.1007/978-1-4614-6534-8](https://doi.org/10.1007/978-1-4614-6534-8).
- [15] Kwang-soo Kim, Wook Bahn, Changhun Lee, Tae-jae Lee, M.M. Shaikh i Kwang-soo Kim. „Vision system for mobile robots for tracking moving targets, based on robot motion and stereo vision information”. W: *IEEE/SICE International Symposium on System Integration*. Japan, Kyoto: IEEE, grud. 2011, s. 634–639. DOI: [10.1109/SII.2011.6147522](https://doi.org/10.1109/SII.2011.6147522).
- [16] Tae-Il Kim, Wook Bahn, Chang-Hun Lee, Tae-Jae Lee, Byung-Moon Jang, Sang-Hoon Lee, Min-Wug Moon i Dong-Il Cho. „A Robotic Pan and Tilt 3-D Target Tracking System by Data Fusion of Vision, Encoder, Accelerometer, and Gyroscope Measurements”. W: *Intelligent Robotics and Applications - 5th International Conference*. Canada, Montreal: Springer Berlin Heidelberg, paź. 2012, s. 676–685. DOI: [10.1007/978-3-642-33515-0_66](https://doi.org/10.1007/978-3-642-33515-0_66).
- [17] Martijn Liem, Arnoud Visser i Frans Groen. „A hybrid algorithm for tracking and following people using a robotic dog”. W: *Proceedings of the 3rd ACM/IEEE international conference on Human robot interaction. HRI '08*. Netherlands, Amsterdam: ACM, 2008, s. 185–192. DOI: [10.1145/1349822.1349847](https://doi.org/10.1145/1349822.1349847).
- [18] David G. Lowe. „Distinctive Image Features from Scale-Invariant Keypoints”. W: *International Journal of Computer Vision* 60.2 (list. 2004), s. 91–110. DOI: [10.1023/B:VISI.0000029664.99615.94](https://doi.org/10.1023/B:VISI.0000029664.99615.94).
- [19] Emilio Maggio i Andrea Cavallaro. *Video Tracking - Theory and Practice*. 1 wydr. John Wiley i Sons, 2011.
- [20] Ivan Marković, François Chaumette i Ivan Petrović. „Moving object detection, tracking and following using an omnidirectional camera on a mobile robot”. W: *IEEE International Conference on Robotics and Automation*. Hong Kong: IEEE, czer. 2014, s. 5630–5635. DOI: [10.1109/ICRA.2014.6907687](https://doi.org/10.1109/ICRA.2014.6907687).
- [21] Miguel A. Olivares-Méndez, Pascual Campoy, Carol Martínez i Iván Mondragón. „A pan-tilt camera Fuzzy vision controller on an unmanned aerial vehicle”. W: *IEEE/RSJ International Conference on Intelligent Robots and Systems*. USA, MO, St. Louis: IEEE, paź. 2009, s. 2879–2884. DOI: [10.1109/IROS.2009.5354576](https://doi.org/10.1109/IROS.2009.5354576).
- [22] Jianbo Shi i Carlo Tomasi. „Good features to track”. W: *Computer Society Conference on Computer Vision and Pattern Recognition (CVPR)*. USA, Washington, Seattle: IEEE, czer. 1994, s. 593–600. DOI: [10.1109/CVPR.1994.323794](https://doi.org/10.1109/CVPR.1994.323794).
- [23] Arnold W. M. Smeulders, Dung M. Chu, Rita Cucchiara, Simone Calderara, Afshin Dehghan i Mubarak Shah. „Visual Tracking: An Experimental Survey”. W: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 36 (list. 2010), s. 1442–1468. DOI: [10.1109/TPAMI.2013.230](https://doi.org/10.1109/TPAMI.2013.230).
- [24] Richard Szeliski. *Computer Vision - Algorithms and Applications*. Springer-Verlag London, 2011.
- [25] Carlo Tomasi i Takeo Kanade. *Detection and Tracking of Point Features*. Spraw. tech. CMU-CS-91-132. USA, Pennsylvania, Pittsburgh: School of Computer Science Carnegie Mellon University, kw. 1991.
- [26] Marco Alexander Treiber. *An Introduction to Object Recognition - Selected Algorithms for a Wide Variety of Applications*. 1 wydr. Springer-Verlag London, 2010.
- [27] Greg Welch i Gary Bishop. *An Introduction to the Kalman Filter*. Spraw. tech. University of North Carolina at Chapel Hill, 1995.
- [28] Alper Yilmaz, Omar Javed i Mubarak Shah. „Object Tracking: A Survey”. W: *ACM Computing Surveys* 38.4 (grud. 2006), s. 1–45. DOI: [10.1145/1177352.1177355](https://doi.org/10.1145/1177352.1177355).

- [29] Zhaoxiang Zhang, Ruhan Sa i Yunhong Wang. „A real time object tracking approach for mobile robot visual servo control”. W: *First Asian Conference on Pattern Recognition (ACPR)*. China, Beijing: IEEE, list. 2011, s. 500–504. doi: [10.1109/ACPR.2011.6166627](https://doi.org/10.1109/ACPR.2011.6166627).