POLITECHNIKA WARSZAWSKA
WYDZIAŁ ELEKTRYCZNY
INSTYTUT STEROWANIA I ELEKTRONIKI PRZEMYSŁOWEJ

PRACA MAGISTERSKA
na kierunku ELEKTROTECHNIKA

Marcin Bzdawski
Nr imm. 192980

Rok. akad. 2007/2008
Warszawa, 8.10.2007

ŚLEDZENIE OBIEKTÓW W SEKWENCJACH OBRAZÓW

Zakres pracy:

1. *Wprowadzenie*
2. *Elementy przetwarzania obrazów w zagadnieniu śledzenia obiektów*
3. *Implementacja i analiza porównawcza wybranych algorytmów śledzenia obiektów*
4. *Modyfikacja wybranego algorytmu śledzenia obiektów*
5. *Podsumowanie i wnioski*

Podpis i pieczątka

Kierownika Zakładu Dydaktycznego

Opiekun naukowy:
Dr inż. Witold Czajewski

Termin wykonania: 15.09.2008

ŚLEDZENIE OBIEKTÓW W SEKWENCJACH OBRAZÓW

Streszczenie

Niniejsza praca porusza temat śledzenia obiektów w sekwencjach obrazów. Przegląd popularnych metod, zamieszczony w pierwszej części pracy znacznie uprościł wybór algorytmów do badań, do których ostatecznie wykorzystano: metodę ciągłej adaptacji przesunięcia (Camshift), dopasowania do szablonu (Template Matching) oraz przepływu optycznego (Optical Flow). Zasadniczym celem pracy było zbadanie skuteczności działania wyżej wymienionych metod śledzenia obiektów. W ramach eksperymentów nagrano kilkanaście sekwencji filmowych, dla których przeprowadzone zostały próby działania powyższych algorytmów. Badania zrealizowane zostały przy wykorzystaniu otwartej biblioteki do przetwarzania obrazów OpenCV, która udostępnia implementacje najbardziej popularnych metod śledzenia obiektów oraz innych funkcji związanych z przetwarzaniem obrazów przydatnych podczas tworzenia pracy.

Testowanie algorytmów nie mogło odbyć się bez określenia modelu badanego obiektu. W tym celu stworzona została funkcja umożliwiająca wybór obiektu, na którego podstawie tworzony był jego określony model (zależny od badanej metody). Następnie dla kolejnych 100 klatek sekwencji wykonywana była lokalizacja obiektu. W ten sposób wyznaczane były takie parametry jak: skuteczność działania algorytmu (wyrażana jako stosunek liczby obrazów, na których obiekt został prawidłowo zlokalizowany do całkowitej liczby obrazów sekwencji) oraz czas przetwarzania (wyrażany jako średni czas lokalizacji obiektu dla wszystkich obrazów sekwencji). W oparciu o przeprowadzone testy można było określić, która z wybranych metod sprawdza się w konkretnych zastosowaniach.

Drugim celem pracy było stworzenie własnego algorytmu śledzenia obiektów w oparciu o wcześniej poznane metody. W pierwszej kolejności określony został sposób wyboru obiektu z obrazu sekwencji, a następnie wybrane zostały cechy wykorzystywane podczas jego śledzenia. Jako podstawy parametr opisujący badany obiekt użyty został histogram składowych obrazu obiektu w różnych przestrzeniach barw. Lokalizacja obiektu w tym wypadku polega na porównaniu histogramu modelu z histogramami fragmentów aktualnej klatki sekwencji. Miejsce, w którym są one najbardziej zbliżone oznacza pozycję obiektu. Trzeba również pamiętać, że histogramy były definiowane na podstawie fragmentów o takich samych rozmiarach, a co za tym idzie porównywane były również ich wymiary. Nie można

także zapomnieć o czynnikach zewnętrznych negatywnie wpływających na wyniki badań takich jak np. zmienne oświetlenie. Zastosowanie histogramów dwuwymiarowych jedynie dla składowych H i S zmniejszyło jego niekorzystny wpływ.

W kolejnym etapie prace objęły zagadnienie eliminacji wpływu na lokalizację obiektu takich zjawisk jak rotacja obiektu, czy zmiany jego odległości względem urządzenia rejestrującego. Udało się to zrealizować poprzez zastosowanie macierzy rotacji w odniesieniu do maski definiującej obrys obiektu.

Ostatnim etapem budowy własnego algorytmu była optymalizacja jego działania pod względem czasu przetwarzania. W pierwotnej wersji model obiektu porównywany był ze wszystkimi fragmentami aktualnej klatki sekwencji. Takie podejście oznaczało maksymalną ilość iteracji podczas procesu lokalizacji obiektu. Aby zmniejszyć ich liczbę zastosowano takie operacje jak: zawężanie okna przeszukiwań, zależność iteracji od wartości dopasowania, likwidacja błędnych skoków w lokalizacji czy metoda Gaussa – Seidla.

Tak stworzony algorytm został przetestowany w taki sam sposób jak metody Camshift, Template Matching oraz Optical Flow. Dało to możliwość odniesienia jego wyników stosunku do badanych algorytmów, a co za tym idzie określenia, czy jest on od nich lepszy czy gorszy. Na podstawie zebranych wyników stwierdzono, że stworzony algorytm wykazał większą skuteczność działania dla większości badanych sekwencji, przy porównywalnym średnim czasie przetwarzania.

OBJECT TRACKING IN IMAGE SEQUENCES

Abstract

This paper addresses tracking of the objects in the image sequences. Review of the popular methods made in the first part of this paper, simplified selection of the algorithms used for testing to the methods of: continuously adaptive mean shift (Camshift), fit the template (Template Matching) and optical flow (Optical Flow). The main aim of this study was to investigate the effectiveness of these methods to track the objects. During the experiments several video sequences were recorded and tests of these algorithms were carried out. These tests were done by using an open source library for image processing OpenCV, which provides implementations of the most popular methods for tracking and other useful functions related to the images processing.

The algorithms testing couldn't be made without a determination of the test object model. For this purpose, the function has been developed for selecting the object on the basis of which was created a specific model (depending on the test method). Then the localization of the object was performed for the next 100 frames sequence. The following parameters were determined: the efficiency of the algorithm (expressed as the ratio of the number of images in which the object was correctly localized to the total number of sequences) and the processing time (expressed as the average localization of the object for all images of the sequence). The tests results allowed to determine, which method should be chosen for the specific applications.

The second objective was to create a custom object tracking algorithm based on the previously known methods. At first an object from the image sequence was chosen, and then the selected features were used during the tracking. The histogram of an object image component in a different color spaces was used as a basic parameter describing the test object. Localization of the object in this case is based on comparison of the model histogram and histograms fragments of a current sequence frame. The place where they are most similar is the position of the object. We must also remember that the histograms were defined on the basis of fragments of the same size, and thus were also compared to their dimensions. Nor can we forget the external factors adversely affecting the results of studies such as variable lighting. Using only two-dimensional histograms for H and S components decreased its negative impact.

The elimination of the issues affecting the localization of the object: the rotation of the object or change its distance relatively to the recording equipment took place in the next stage. This has been achieved by applying a rotation matrix with respect to the mask that defines the outline of the object.

The last step was to build specific algorithm to optimize its operation in terms of processing time. In the original version of the object model was compared with all parts of the current sequence frame. This approach meant the maximum number of iterations during the process of object localization. To reduce the number of such operations was used: the window narrowing searches, the dependence of the fitting iteration, the elimination of erroneous jumps in the localization or method of Gauss - Seidel.

That created an algorithm which has been tested in the same manner as the method Camshift, Template Matching and Optical Flow. This gave the opportunity to reference the results to test the algorithms, and thus determine whether it is better or worse than algorithms. Based on the collected results, the developed algorithm has demonstrated greater efficacy for the majority of the sequences, with comparable average time of processing.

Warszawa, dnia ..... roku.

Politechnika Warszawska
Wydział Elektryczny

OŚWIADCZENIE

Świadom odpowiedzialności prawnej oświadczam, że niniejsza praca dyplomowa magisterska pt. „ŚLEDZENIE OBIEKTÓW W SEKWENCJACH OBRAZÓW ”

- została napisana przeze mnie samodzielnie
- nie narusza niczyich praw autorskich
- nie zawiera treści uzyskanych w sposób niezgodny z obowiązującymi przepisami.

Oświadczam, że przedłożona do obrony praca dyplomowa nie była wcześniej podstawą postępowania związanego z uzyskaniem dyplomu lub tytułu zawodowego w uczelni wyższej.

Jestem świadom, że praca zawiera również rezultaty stanowiące własności intelektualne Politechniki Warszawskiej, które nie mogą być udostępniane innym osobom i instytucjom bez zgody Władz Wydziału Elektrycznego.

Oświadczam ponadto, że niniejsza wersja pracy jest identyczna z załączoną wersją elektroniczną.

Imię i Nazwisko dyplomanta: Marcin Bzdawski
Podpis dyplomanta: .....

Spis treści

1 Wprowadzenie

W obecnym świecie rejestracja obrazów jest coraz powszechniejsza. Nie dziwią nas już kamery w supermarketach, bankach, czy na ulicy. Dzięki coraz większej dostępności wszelkiego rodzaju urządzeń, takich jak aparaty, czy kamery cyfrowe, my także możemy rejestrować otaczający nas świat. Otrzymane w ten sposób obrazy, czy ich sekwencje, mogą mieć różne zastosowania. Dla większości ludzi jest to po prostu zapis obrazu z danej chwili w postaci cyfrowej, który można w późniejszym czasie odtworzyć. Jednak dla naukowców ważne jest to, że każdy taki obraz, czy sekwencję obrazów można odpowiednio przetworzyć i poddać dokładnej analizie, aby wydobyć z niej istotne informacje.

Ze względu na to, że otrzymane sekwencje mogą być bardzo złożone, tworzone są automatyczne metody analizy sekwencji. Jednym z zagadnień z tym związanych jest problem śledzenia obiektów w sekwencjach obrazów. Dla człowieka jest to zadanie bardzo proste ze względu na jego bardzo dobre rozumienie obrazów. Jednak dla maszyny jest to proces skomplikowany, gdyż dla niej obraz jest zbiorem pikseli, który musi dopiero zostać poddany procesowi obróbki, aby w efekcie otrzymać wymagane informacje. Oczywiście zaimplementowanie ludzkiego poziomu rozumienia byłoby rozwiązaniem optymalnym, lecz ze względu na to, że nie wiadomo, jak dokładnie działa ludzki mózg, nie jest to możliwe. Z tego powodu powstał szereg metod, umożliwiających w lepszy lub gorszy sposób śledzenie obiektów w sekwencjach obrazów. Wybrane metody z tego zakresu oraz proces tworzenia jednej z nich są tematem tej pracy.

1.1 Cel pracy

Celem niniejszej pracy jest przegląd zagadnień związanych z problemem śledzenia obiektów w sekwencjach obrazów, weryfikacja działania wybranych metod śledzenia obiektów oraz stworzenie własnego algorytmu w oparciu o opisane metody. Głównym narzędziem wykorzystywanym w pracy jest stworzona przez firmę INTEL CORPORATION biblioteka OpenCV (ang. Open Source Computer Vision Library). Została ona wybrana ze względu na szeroki zasób funkcji związanych zarówno z przetwarzaniem obrazów (również z zagadnieniem śledzenia obiektów) jak i widzeniem komputerowym. Istotne jest również to, że jest darmową biblioteką środowisk programistycznych języka C o otwartym kodzie źródłowym, co w dużym stopniu ułatwia korzystanie z jej funkcjonalności. [1]

1.2 Układ pracy

Niniejsza praca magisterska zawiera zarówno część opisową, wprowadzającą w dziedzinę przetwarzania obrazów jak również praktyczną, w której oprócz przykładów zastosowań wybranych funkcji oraz oceny ich działania, umieszczony jest opis tworzenia własnego algorytmu śledzenia obiektów w sekwencjach obrazów. Praca podzielona jest na rozdziały zawierające:

Rozdział 2: wprowadzenie w tematykę przetwarzania obrazów, opis narzędzi potrzebnych do poprawy ich jakości oraz wyciągnięcia z nich ważnych elementów ułatwiających ich dalszą analizę;

Rozdział 3: przegląd wybranych (gotowych) algorytmów śledzenia obiektów w sekwencjach obrazów, przeprowadzone testy oraz ocenę ich możliwości, zalety i wady;

Rozdział 4: opis procesu tworzenia algorytmu śledzenia obiektów w sekwencjach obrazów wraz z przykładami jego działania;

Rozdział 5: spostrzeżenia autora odnośnie tematyki przetwarzania obrazów w szczególności zastosowaniu jej do śledzenia obiektów w sekwencjach obrazów, opis napotkanych trudności, które udało się rozwiązać oraz indywidualny wkład autora w napisanie pracy magisterskiej;

2 Elementy przetwarzania obrazów w zagadnieniu śledzenia obiektów

2.1 Wprowadzenie

Dziedzina przetwarzania obrazów jest znana od kilkudziesięciu lat. Ostatnimi czasy stała się coraz bardziej popularna, dzięki coraz większej dostępności urządzeń takich jak aparaty czy kamery cyfrowe, oraz wszechobecnej komputeryzacji . Można ją spotkać m.in. w systemach monitoringu w hipermarketach, przy identyfikacji tożsamości np. na podstawie linii papilarnych, w fabrykach przy kontroli, jakości w medycynie do analizy zdjęć tomograficznych oraz w wielu innych zastosowaniach.

Przetwarzanie obrazów składa się z wielu różnych etapów i operacji takich jak: [2]

- akwizycja;
- przetwarzanie wstępne;
- binaryzacja;
- segmentacja;
- analiza i etykietyzacja;
- rozpoznanie i interpretacja wyników;

Pierwszym i zarazem podstawowym etapem przetwarzania obrazów jest akwizycja obrazu (sekwencji obrazów), czyli uzyskanie jego odpowiedniej postaci najczęściej cyfrowej. Jest to realizowane na dwa sposoby: odczyt z systemu wizyjnego w czasie rzeczywistym oraz zapis / odczyt z pliku. Otrzymana postać powinna mieć odpowiednie, zgodne z wymaganiami parametry jak np.: rozdzielczość, zakres kolorów, format itd. [3]

Często jednak otrzymane obrazy nie są pozbawione błędów, nie spełniają przyjętych wymagań. Istotna staje się odpowiednia ich obróbka, czyli przetwarzanie wstępne tak jak poprawa ich jakości. Można to osiągnąć m.in. poprzez zastosowanie różnych metod wstępnej obróbki obrazu, do których zaliczają się metody

morfologiczne, filtracja itp., które dokładniej zostały opisane w kolejnych rozdziałach. Ułatwia to prawidłowe przeprowadzenie kolejnych procesów przetwarzania obrazów.

Etap binaryzacji jest często pomijany lub wykorzystywany w kolejnych krokach. Polega on na konwersji obrazu do postaci zero-jedynkowej wedle określonego kryterium. Najprostszym przykładem binaryzacji jest progowanie, które polega na przydzielenie wartości 0 pikselom z określonego zakresu barw, natomiast 1 wszystkim pozostałym. W wyniku uzyskuje się obraz czarno biały. [2]

Segmentacja jest operacją polegającą na wyodrębnieniu z obrazu źródłowego elementów charakteryzujących się pewnymi unikatowymi dla niego własnościami Mogą nimi być np.: kolor, kształt, wymiary, wygląd itd. Na obrazie wynikowym pozostawione zostają tylko te fragmenty, które spełniają przyjęte kryteria. [4] Wraz z rozwojem techniki powstało wiele metod segmentacji obrazów, z których można wyróżnić:

- metody punktowe – obszary tworzone są na podstawie cech przypisanych do pikseli. Zaliczamy do nich progowanie czy klasteryzację;
- metody krawędziowe – obszary tworzone są na podstawie wcześniej uzyskanego obrazu konturowego;
- metody obszarowe – zaliczamy do nich rozrost obszaru, łączenie obszarów czy segmentację wododziałową;
- metody morfologiczne – wykorzystują one podstawowe oraz złożone operacje morfologiczne;

W kolejnym etapie analizy określane są cechy obiektów, fragmentów obrazu wyodrębnionych w procesie segmentacji. Do cech można zaliczyć m.in. współczynniki niezmiennicze względem przekształceń obrazów (tak jak obrotu, przesunięcia, zmiany, skali), cechy topologiczne (tak jak liczba otworów), cechy szkieletowe (tak jak zakończenia szkieletu, rozwidlenia szkieletu), kolor itp. Wszystkie cechy mają zwykle postać liczb i przetrzymywane są w wektorze cech. Istotne jest, aby uzyskany wektor cech dla danego obiektu był jak najbardziej unikatowy w stosunku do pozostałych obiektów. W ten sposób będzie można dużo lepiej odróżnić od siebie poszczególne obiekty. Ważne jest również oznaczenie pozycji obiektów oraz przynależności do nich poszczególnych pikseli obrazu, co jest

uzyskiwane poprzez ich etykietyzację. Polega ona na oznaczeniu przynależności pikseli do obiektów, poprzez przydzielenie każdemu z nich etykiety od 1 do n (każda z nich oznaczająca jeden obiekt). Jako wynik otrzymywany jest obraz, na którym każdy piksel zawiera wartość etykiety obiektu, do którego został przyporządkowany.

Na ostatnim etapie rozpoznania określana jest przynależność otrzymanych w procesie analizy cech do wcześniej zdefiniowanej klasy. Jest to tzw. klasyfikacja, natomiast algorytm za nią odpowiadający nazywany jest klasyfikatorem. Aby klasyfikator prawidłowo klasyfikował cechy musi wcześniej zostać odpowiednio przygotowany – „nauczony". Jest to najczęściej realizowane poprzez wyznaczenie parametrów klasyfikatora, czyli porównaniu wektorów cech o znanych wynikach klasyfikacji zbioru uczącego ze zbiorem testowym o takim samym charakterze. W optymalnym przypadku wszystkie wektory cech z zbioru testowego powinny zostać poprawnie sklasyfikowane. [5]

Poniżej zamieszczono opis wybranych zagadnień z dziedziny przetwarzania obrazów, istotnych z punku widzenia zagadnienia śledzenia obiektów w sekwencjach obrazów oraz przydatnych podczas pisania pracy.

2.2 Morfologia matematyczna

Morfologia matematyczna została zainicjowana przez H. Minkowskiego na początku XX wieku. Można jednak przyjąć, że jej geneza przypada na lata 60-te i miała miejsce w Centre de Morphologie Mathematique w Paris School of Mines w Fontainblean, a jej prekursorami byli G. Matherno oraz J. Serra, których podstawowa koncepcja oparta była na nieobiektywności struktury geometrycznej. Według nich jej pełny opis można poznać dopiero przy wykorzystaniu „elementu strukturalnego". Opisując to zagadnienie, swoje koncepcje oraz podstawowe operacje na obrazie przyczynili się do dużego rozwoju nauki. Podstawową zaletą morfologii matematycznej jest dosyć prosty zapis operacji na obrazie, jak i również ich realizacji. [6] , [7]

2.2.1 *Podstawowe operacje morfologiczne*

Podstawowymi operacjami morfologicznymi nazywamy takie czynności, których nie da się zapisać w prostszej postaci np. poprzez inne operacje morfologiczne. Zaliczamy do nich [7], [8]: infimum, supremum, różnica obrazów, progowanie, dylację, erozję. Oczywiście jest ich o wiele więcej, ale ze względu na rozległość tego tematu skupie się jedynie na tych, które stanowią podstawę morfologii matematycznej, a ich zastosowanie jest przydatne w procesie przygotowywania obrazu np. do ekstrakcji cech.

Infimum

Infimum dwóch obrazów f_1 i f_2 w punkcie $(x; y)$ stanowi ta z funkcji f_1 i f_2, której wartość dla piksela jest mniejsza:

$$(f_1 \cap f_2)(x; y) = \min\_f_1(x; y); f_2(x; y)$$

gdzie:

$f_1(x; y)$, $f_2(x; y)$ – wartości poziomów szarości pikseli dwóch obrazów wejściowych.

Supremum

Supremum dwóch obrazów f_1 i f_2 w punkcie $(x; y)$ stanowi ta z funkcji f_1 i f_2, której wartość dla danego piksela jest większa:

$$(f_1 \cup f_2)(x; y) = \max\_f_1(x; y); f_2(x; y)$$

gdzie:

$f_1(x; y)$, $f_2(x; y)$ – wartości poziomów szarości pikseli dwóch obrazów wejściowych.

Różnica

Różnicę obrazów opisanych funkcjami f_1 i f_2 definiujemy jako przekształcenie, w wyniku którego powstaje obraz o wartościach pikseli odpowiadających różnicy pikseli obrazów f_1 i f_2. Zależność tą można przedstawić następująco:

$$(f_1 - f_2)(x; y) \begin{cases} f_1(x; y) - f_2(x; y), & \text{jeśli } f_1(x; y) > f_2(x; y) \\ 0, & \text{jeśli } f_1(x; y) \leq f_2(x; y) \end{cases}$$

Progowanie

Operacja ta jest jedną z najprostszych metod, dzięki którym można wydzielić jakiś obszar z obrazu. Opiera się ona na porównywaniu wartości pikseli z przyjętym progiem. Najprostsza sytuacja ma miejsce, gdy mamy tylko jedną wartość progu tak jak np.:

$$O_p(x,y) = \begin{cases} 1, gdy\ O(x,y) > p \\ 0, gdy\ O(x,y) \leq p \end{cases}$$

gdzie:
p – wartość progu;

Powyższy przykład jest bardzo prosty, ale w dosyć klarowny sposób pokazuję działanie progowania. Stosując powyższe równania na obrazie w odcieniach szarości uzyskiwany jest obraz binarny.

Oczywiście działanie progowania może być dużo bardziej skomplikowane i jest zależne od oczekiwanego efektu. Często wykorzystywane są równania wielo-progowe, operujące nie tylko na wartościach pikseli, ale także histogramów zarówno w odniesieniu do całego obrazu (globalnie) jak i jego fragmentów (lokalnie).

Dylacja

Dylacja obrazu A przez element strukturalny B jest zdefiniowana jako suma przesunięć zbioru A o wszystkie elementy b, przy czym b ε B

$$A \oplus_s B = \bigcup_{b \varepsilon B} A_b$$

Przy operacji dylacji warto wprowadzić pojęcie elementu strukturalnego, czyli powierzchni pikseli, która będzie sprawdzana podczas jednej iteracji. Jest on siatką składającą się z określonej liczby elementów, do których przyporządkowane są odpowiednie punkty badanego obrazu. Zależnie od wykonywanej operacji zmieniane są ich wartości (głównie środkowa).

Na podstawie pojęcia elementu strukturalnego operację dylacji określamy jako jego B - przemieszczanie po wszystkich pikselach obrazu A. Wartość każdego z

punktów obrazu wyjściowego odpowiada sumie logicznej elementu strukturalnego oraz odpowiedniego punktu badanego obrazu, który jest nim „przysłonięty”.

Przykład:

a) b)

Rysunek 1. a) obraz oryginalny, b) obraz po wykonaniu na nim operacji dylacji

Jak widać operacja dylacji rozszerza obszary, które są jasne zawężającym tym samym te, które są ciemne.

Erozja

Erozja obrazu A przez element strukturalny B jest zdefiniowana, jako przecięcie przesunięć zbioru A o wszystkie elementy b, przy czym b ε B

$$A \ominus_s B = \bigcup_{b \varepsilon B} A_b$$

Jej działanie jest identyczne jak operacji dylacji z tą jedyną różnicą, że wartości pikseli obrazu wynikowe nie są sumą, lecz iloczynem elementu strukturalnego oraz odpowiedniego punktu badanego obrazu, który jest nim „przysłonięty”.

Przykład:

a) b)

Rysunek 2. a) obraz oryginalny, b) obraz po wykonaniu na nim operacji erozji

Jak widać operacja erozji rozszerza obszary, które są ciemniejsze zawężając przy okazji jaśniejsze.

2.2.2 *Złożone operacje morfologiczne*

Złożone operacje morfologiczne to takie czynności, których definicja jest złożeniem kilku operacji podstawowych. Zaliczamy do nich m.in.: otwarcie, zamknięcie, gradient morfologiczny oraz wiele innych. [7], [8]

Otwarcie

Otwarcie jest operacją powstałą z zastosowania dylacji na obrazie wcześniej poddanym działaniu erozji.

$$A \circ_s B = [A \ominus_s B] \oplus_s B$$

Przykład:

a) b)

Rysunek 3. a) obraz oryginalny, b) obraz po wykonaniu na nim operacji otwarcia

Jak widać na powyższym przykładzie, elementy ciemne takie jak czułki motyla prawie znikły z obrazu. Również cienkie niepołączone na skrzydłach linie uległy rozjaśnieniu, przez co przerwa pomiędzy ich końcem, a dołem skrzydeł powiększyła się.

Zamkniecie

Zamknięcie jest operacją powstałą z zastosowania erozji na obrazie wcześniej poddanym działaniu dylacji.

$$A \bullet_s B = [A \oplus_s B] \ominus_s B$$

Przykład:

a) b)

Rysunek 4. a) obraz oryginalny, b) obraz po wykonaniu na nim operacji zamknięcia

Jak widać na powyższym przykładzie operacja zamknięcia spowodowała połączenie się czarnych zarysów na skrzydłach. Jest to bardzo dobry sposób poprawy jakości obrazów konturowych.

Gradient morfologiczny

Gradient morfologiczny jest operacją, której wynik jest różnicą dylacji i erozji wykonanych na zadanym obrazie.

$$\gamma = (A \oplus_s B) - (A \ominus_s B)$$

Przykład:

a) b)

Rysunek 5. a) obraz oryginalny, b) obraz po wykonaniu na nim operacji gradientu morfologicznego

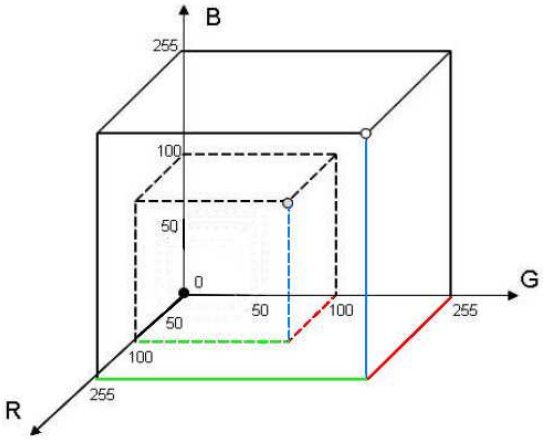
Na powyższym przykładzie widać, że zastosowanie operacji gradientu morfologicznego jest jednym z najprostszych sposobów uzyskania obrazu konturowego znajdujących się na nim elementów.

2.3 Przestrzenie barw

Bardzo istotnym aspektem przetwarzania obrazów jest pojęcie przestrzeni barw. Ich pierwszy model został opracowany już w 1920 r. przez Commission International de d'Eclairage – „CIE" i zawierał cały zakres widma widzialnego. [9] Wraz z rozwojem techniki powstawały coraz to nowe sposoby reprezentacji kolorów, z których najpopularniejszym stał się model *RGB* (ang. Red, Green, Blue) składający się z trzech barw podstawowych czerwonej, zielonej i niebieskiej. Każda z nich przyjmuje wartość z zakresu od 0 do 255, co daje możliwość zakodowania aż: [6]

$$256 * 256 * 256 = 16.7\ mln\ kolorów$$

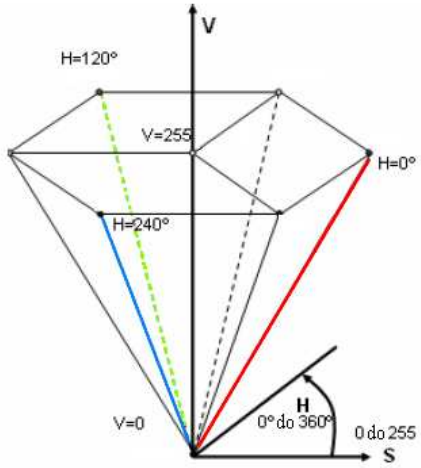
Kolor piksela zakodowany w tym modelu posiada trzy wartości odpowiednio umiejscowione w przestrzeni barw.



Rysunek 6. Przestrzeń barw RGB.

Model RGB jest najprostszym sposobem zakodowania tak dużej palety barw. Według przyjętego przez CIE w 1931 r. opisu jest on tworzony przy użyciu fal monochromatycznych kolorów czerwonego, zielonego i niebieskiego o długościach 700 nm, 546.1 nm i 435.8 nm. Norma europejska EBU określa składowe RGB w oparciu o światło emitowane przez fosfor. Poszczególne punkty obrazu przyjmują jedną z trzech barw a poprzez działanie uśredniające oka ludzkiego widziany jest odpowiedni kolor. Jest to wykorzystywane np. w telewizorach czy monitorach komputerowych w lampach kineskopowych. Oprócz tego model RGB stosowany jest aparatach, kamerach, projektorach itd. [3], [9]

Kolejnym wartym uwagi modelem barw jest model HSV (ang. Hue, Saturation, Value), w którym kolejne składowe odpowiadają barwie, jej nasyceniu oraz jasności. Jest przestrzenią biegunową, której zasada konstrukcji oparta jest na sposobie mieszania farb przez malarzy. Pierwsza składowa reprezentuje kolor co jest odmienne w stosunku do modelu RGB, gdzie było to zależne aż od trzech składowych. Poniżej pokazano model barw HSV z oznaczeniem kolorów podstawowych RGB.



Rysunek 7. Przestrzeń barw HSV

Obraz zapisany w modelu RGB można po odpowiednich przeliczeniach przedstawić w postaci składowych HSV. Ich wartości obliczane są z poniższych wzorów:

$$H = \begin{cases} \dfrac{G-B}{\max(R,G,B) - \min(R,G,B)} \; gdy\; R = \max(R,G,B) \\ 2 + \dfrac{B-R}{\max(R,G,B) - \min(R,G,B)} \; gdy\; G = \max(R,G,B) \\ 4 + \dfrac{R-G}{\max(R,G,B) - \min(R,G,B)} \; gdy\; R = \max(R,G,B) \end{cases}$$

$$S = \begin{cases} 0 \; gdy \max(R,G,B) = 0 \\ \dfrac{\max(R,G,B) - \min(R,G,B)}{\max(R,G,B)} \; gdy \max(R,G,B) > 0 \end{cases}$$

$$V = \max(R,G,B)$$

Możliwe jest również przejście z przestrzeni HSV do RGB. Jest to proste, gdy S=0, gdyż wtedy wszystkie składowe RGB = V. W przeciwnym wypadku zakres składowej H dzielony jest na sześć przedziałów, a następnie dla każdego z nich poszczególne wartości wyznaczane są wedle poniższych zależności:

$$(R,G,B) = \begin{cases} (max, mid_1, min), gdy\ i=0 \\ (mid_2, max, \min), gdy\ i=1 \\ (min, max, mid_1), gdy\ i=2 \\ (min, mid_2, \max), gdy\ i=3 \\ (mid_1, min, \max), gdy\ i=4 \\ (max, min, mid_2), gdy\ i=6 \end{cases}$$

gdzie:

i – przedział H;

min– V(1-S);

mid_1 – V(1-S(H-i));

mid_2 – V(1-S(1-H+i));

max – V;

Model przestrzeni HSV jest bardziej intuicyjny dla człowieka ze względu na występowanie składowej V określającej jaskrawość, która ma bardzo duży wpływ na postrzeganie kolorów przez oko ludzkie [3]. Dzięki takiemu zapisowi poszczególnych kolorów model ten znalazł zastosowanie w grafice komputerowej [9]. Na Rysunku 8. można zobaczyć obraz w przestrzeni RGB wraz z odpowiadającymi mu obrazami składowych R, G, B i H, S, V. Bardzo popularne są również odmiany modelu HSV takie jak IHSV, HLS czy IHLS różniące się jedynie zakresami poszczególnych składowych.

Istnieją także inne, rzadziej stosowane modele barw takie jak CMY (ang. cyan, magenta, yellow),czy CMYK, który jest komplementarny do RGB, a zmienione zostały jedynie wartości odpowiednich kolorów. Znalazły one zastosowanie m.in. w poligrafii czy w drukarkach atramentowych. Model CMYK jest rozszerzony o stałą K reprezentującą ilość potrzebnego barwnika czarnego.

a)

b) c) d)

e) f) g)

Rysunek 8.a) obraz w przestrzeni RGB, odpowiadające obrazy poszczególnych składowych b) R, c) G d) B w przestrzeni RGB oraz e) H, f) S, g) V w przestrzeni HSV

2.4 Histogram

Pojęcie histogramu określa częstość występowania poszczególnych poziomów jasności w obrazie cyfrowym. Najczęściej jego wynik przedstawiany jest na wykresie lub w tabeli, zależnie od potrzeb. Rozkład kolorów jest charakterystyczny dla każdego obrazu, dlatego stanowi jedną z jego podstawowych cech.

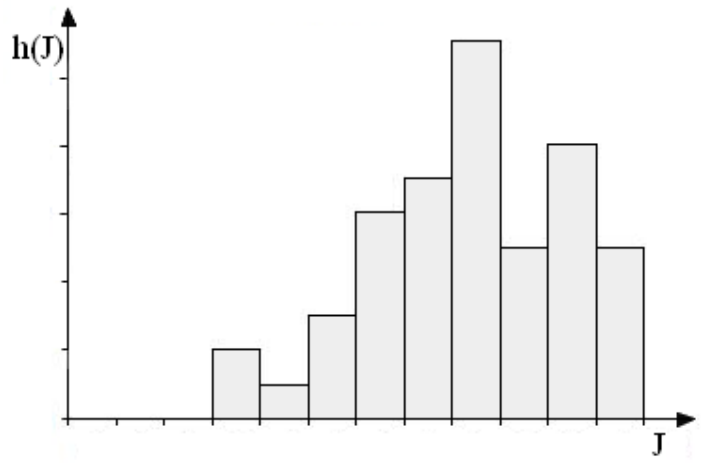
Przyjmując, że J_0, J_1, … ,J_{i-1} określają kolejne „i" możliwych wartości pikseli lub ich zakresów, wtedy odpowiadające im rzędne histogramu $h(J_0),h(J_1)\ldots\ h(J_{i-1})$ przedstawiają ich udział w obrazie. [10]

$$h(J_j) = n_j \quad j = 0, 1, 2 \ldots \ldots i - l$$

gdzie

n_j – liczba pikseli o jasności J_j;

Przykładowy histogram ilustruje poniższy rysunek gdzie na osi odciętych są wartości lub zakresy jasności pikseli J, natomiast na osi rzędnych odpowiadająca ich ilość występowania na obrazie h(J).



Rysunek 9. Przykładowy histogram

Poszczególne składowe histogramu obliczamy ze wzoru:

$$n_j = \sum_{x=0}^{M-1} \sum_{y=0}^{N-1} j_j(x, y) \quad j = 0, 1, 2 \ldots \ldots i - l$$

M-1 N-1

Oprócz histogramów jednowymiarowych jak powyższy, stosowane są również histogramy wielowymiarowe. Podstawową różnicą jest to, że przynależność danego piksela do zakresu zależy nie od wartości jednej składowej, lecz od ich kombinacji. Przykładowo, jeśli uznamy, że zakres histogramu dwuwymiarowego dla składowych H i S wynosi np. H: 0 – 60, S – 60 to piksel o wartości (H=10,S=10) znajduje się w tym przedziale, natomiast piksel o wartości (H=10, S=70) już się nie znajduje. [11]

Przykład histogramu wielowymiarowego został przestawiony w rozdziale 5.3.2 na Rysunku 85.

2.5 Filtracja

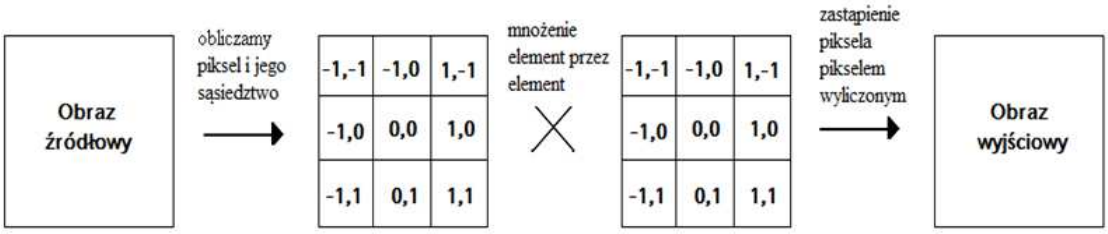
Filtracja obrazów jest nieodłączną częścią dziedziny przetwarzania obrazów. Ze względu na przeznaczenie oraz wymaganą dokładność jej działanie jest zdecydowanie różne. Głównym jej zastosowaniem jest poprawa jakości obrazów poprzez np. likwidację szumów czy poprawę kontrastu. Po zastosowaniu filtracji otrzymywany jest nowy obraz, przekształcony w stosunku do wzorcowego, przy czym do wyznaczenia wartości pikseli obrazu wynikowego potrzebne są informacje o wielu pikselach obrazu źródłowego. [6]

Filtry są głównie stosowane do:

- zlikwidowania, zmniejszenia niepożądanego szumu;
- uwypuklenia cech obrazu, które są dla nas istotne;
- poprawy ostrości obrazu;
- usunięcia zbędnych elementów, wad w obrazie;
- poprawie parametrów technicznych;
- rekonstrukcji obrazów, które uległy zniszczeniu;

2.5.1 Splot

Splot jest operacją polegającą na wyznaczaniu wartości pikseli na podstawie pikseli z nimi sąsiadującymi, przy czym wpływ każdego z nich określa tablica zwana maską splotu. Składa się ona z elementów zwanych współczynnikami splotu. Poniższy rysunek ilustruje działanie tej operacji.



Rysunek 10. Schemat działania splotu

Współczynniki maski są różne dla każdego filtru i zależne od zamierzonego efektu wyjściowego. Do filtrów splotowych możemy zaliczyć np.: [10]

- filtr dolnoprzepustowy– filtry te są stosowana do redukcji szumów, ale również rozmywają krawędzie, przepuszczają elementy o niskiej częstotliwości, natomiast tłumią te o wysokiej częstotliwości;

a)

1	1	1
1	1	1
1	1	1

b)

1	1	1
1	4	1
1	1	1

c)

1	2	1
2	4	2
1	2	1

Rysunek 11. Przykładowe maski dla filtru dolnoprzepustowego

- filtr górnoprzepustowy – filtry te wyostrzają obraz, ale również wzmacniają szumy, odwrotnie do filtrów dolnoprzepustowych wzmacniają elementy o dużej częstotliwości natomiast tłumią te o niskiej częstotliwości;

a)

-1	-1	-1
-1	9	-1
-1	-1	-1

b)

1	-2	1
-2	5	-2
1	-2	1

c)

0	-1	0
-1	20	-1
0	-1	0

Rysunek 12. Przykładowe maski dla filtru górnoprzepustowego

Oprócz powyższych występują również filtry wykrywające krawędzie (konturowe), do których możemy zaliczyć:

- filtr gradientowy kierunkowy – wykrywa krawędzie poprzez rozpoznanie zmiany jasności w danym kierunku;

a)

-1	1	1
-1	-2	1
-1	1	1

b)

-1	-1	1
-1	-2	1
1	1	1

c)

-1	-1	-1
1	-2	1
1	1	1

Rysunek 13. Przykładowe maski dla filtru gradientowego kierunkowego

- filtr Sobela – wykorzystuje pierwsze pochodne, dzięki czemu często daje lepsze wyniki niż filtry dolno i górnoprzepustowy, wykonywane są w tym wypadku dwa przebiegi dla konturów poziomych jak i pionowych;

a)

1	2	1
0	0	0
-1	-2	-1

b)

2	1	0
1	0	-1
0	-1	-2

c)

1	0	-1
2	0	-2
1	0	-1

Rysunek 14. Przykładowe maski dla filtru Sobela

- filtr Prewitta – działanie tego filtru jest bardzo zbliżone do filtru Sobela, różni się stosowanymi maskami;

a)

-1	-1	-1
0	0	0
1	1	1

b)

0	-1	-1
1	0	-1
1	1	0

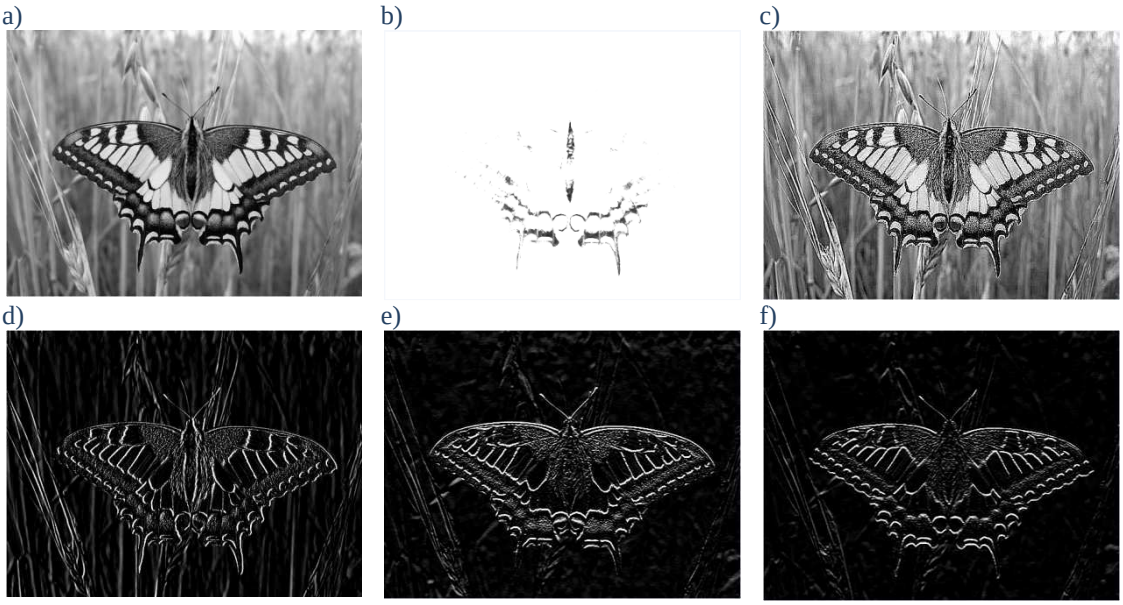
c)

1	0	-1
1	0	-1
1	0	-1

Rysunek 15. Przykładowe maski dla filtru Prewitta

Na poniższym rysunku przedstawione zostało działanie filtrów splotowych. Rysunek 16.a przedstawia przykładowy obraz, który następnie został przefiltrowany:

- filtrem dolnoprzepustowym (z maską Rysunek 11.b);
- filtrem górnoprzepustowym (z maską Rysunek 12.a);
- filtrem gradientowym kierunkowy (z maską Rysunek 13.a);
- filtrem Sobela (z maską Rysunek 14.a);
- filtrem Prewitta (z maską Rysunek 15.a);

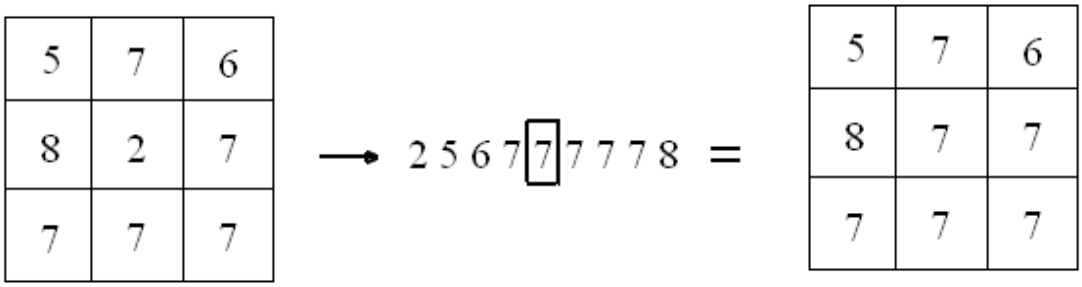


Rysunek 16. Przykład zastosowania dla obrazu a) różnych rodzajów filtrów: b) dolnoprzepustowego, c) górnoprzepustowego, d) gradientowy kierunkowego, e) Sobela, f) Prewitta

Jak widać na powyższym rysunku zastosowanie filtru dolnoprzepustowego zostawiło jedynie najbardziej jaskrawe elementy obrazu, natomiast filtr górnoprzepustowy wyostrzył elementy tła (kontury zboża), lecz jednocześnie zniekształcił elementy motyla. Filtry konturowe wyznaczyły prawidłowo obrys motyla, przy czym najlepszy z nich okazał się filtr Sobela, który rozpoznał najwięcej szczegółów. Kontury elementów tła najlepiej wyznaczył filtr gradientowy kierunkowy, lecz krawędzie na obrazie wynikowym są najbardziej rozmyte.

2.5.2 Filtry medianowe

Zasada działania filtrów tego typu polega na zastępowaniu piksela centralnego maski powstałej z fragmentu obrazu badanego, jej wartością środkową pod względem jasności [6]. Jego działanie ilustruje poniższy rysunek.



Rysunek 17. Schemat działania filtru medianowego

Filtry te bardzo dobrze sprawdzają się przy likwidacji lokalnych szumów tzn., gdy w obrazie znajdują się piksele odbiegające wartością od innych w swoim otoczeniu.

Efekt filtracji medianowej obrazu z szumem impulsowym ilustruje Rysunek 18. Jak widać filtr usunął znaczną część szumów znajdujących się na obrazie źródłowym. Istotną zaletą filtru medianowego jest fakt, że nie rozmazuje krawędzi czy szczegółów obrazu, co występuje w przypadku filtru uśredniającego. Jest często wykorzystywany we wstępnym procesie filtracji. [12]

Przykład:

a) b)

Rysunek 18. a) obraz zaszumiony szumem Sól i Pieprz, b) obraz odszumiony filtrem medianowym

3 Metody śledzenia obiektów

3.1 Wprowadzenie

Podstawą do tworzenia algorytmów śledzenia obiektów jest wykrywanie ruchu. Do tego celu wykorzystywane są zarówno techniki radiowe jak i laserowe, czy wizyjne. W pracy opisane zostały jedynie te ostatnie, które w ostatnich latach stają się coraz bardziej popularne. Jest to spowodowane coraz większą dostępnością wymaganych do jej zastosowania urządzeń (tak jak kamera, aparat czy komputer) oraz odpowiedniego oprogramowania. W tym wypadku obiekt badań przedstawiany jest w postaci sekwencji obrazów, a zadaniem algorytmu śledzenia jest wyznaczenie jego pozycji. Powstało wiele metod to realizujących i można je podzielić na: [13]

- metody przesunięcia regionów (śledzenie regionów);
- metody wykorzystujące kontur obiektu (aktywne śledzenie konturów);
- metody wykorzystujące cechy obiektu (śledzenie obiektów na podstawie ich cech);
- metody oparte na modelu obiektu (śledzenie na podstawie modelu obiektu);

3.2 Śledzenie obszarów (ang. Region-Based Tracking)

Metoda ta polega na sprawdzaniu kolejnych klatek sekwencji obrazów w poszukiwaniu zmian. Znalezione obszary mogą być oznaczane lub też wyznaczany jest ich wektor przesunięcia. Dzięki temu powstały takie algorytmy jak metoda różnicowa czy przepływ optyczny.

3.2.1 Metoda różnicowa (ang. Differential Method)

Jest to metoda, dzięki której uzyskiwany jest obraz różnicowy, na którym widoczne są jedynie elementy poruszające się. Powstaje on na zasadzie odjęcia od siebie dwóch kolejnych klatek obrazu. Obszary, w których obraz nie uległ zmianie zostają wyzerowane, natomiast te, w których wystąpił ruch zostają widoczne. Istotne jest ustalenie wartości progu, z którym porównywane jest przemieszczenie odpowiednich pikseli, gdyż eliminuje to niewielkie ruchy wynikające z szumu. Opisuje to poniższa zależność: [14]

$$r(i,j) = \begin{cases} 0, & gdy\ w_1(i,j) - w_2(i,j) \leq \varepsilon \\ 1, & gdy\ w_1(i,j) - w_2(i,j) > \varepsilon \end{cases}$$

gdzie:

r – obraz różnicowy;

w_1 – piksel obiektu poruszającego się;

w_2 – piksel tła (elementów nie poruszających się);

Metoda ta jest dosyć prosta i najlepiej sprawdza się na obrazach binarnych czy w odcieniach szarości. Poniżej przedstawiono przykład jej działania.

a)

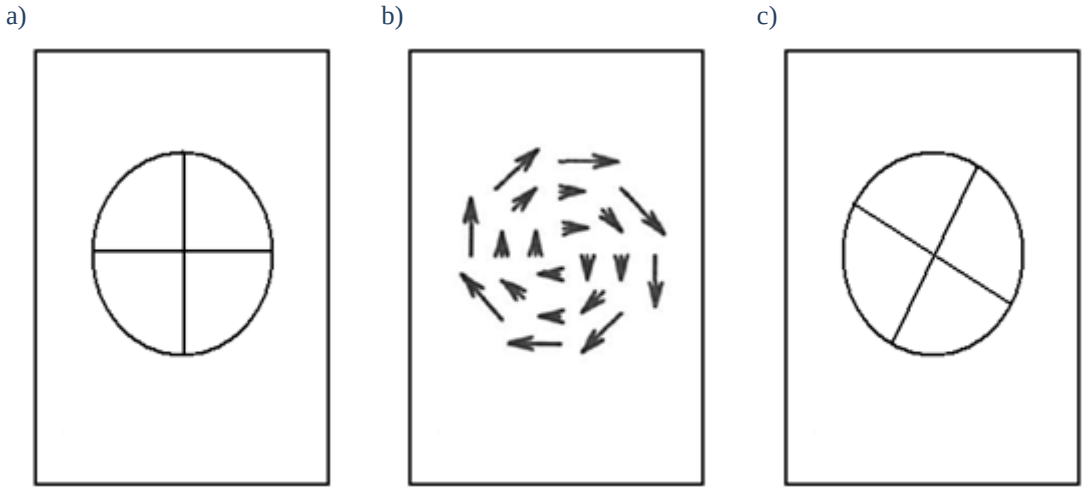
b)

c)

Rysunek 19. a) pierwsza klatka sekwencji obrazów, b) druga klatka sekwencji obrazów, c) obraz różnicowy pierwszej i drugiej klatki

3.2.2 Przepływ optyczny (ang. Optical Flow)

Przepływ optyczny jest metodą wyznaczającą pole wektorowe opisujące ruch obiektu. Kolejny obraz z sekwencji uzyskiwany jest z przesunięcia poszczególnych pikseli obrazu zgodnie z wartościami odpowiednich wektorów, co ilustruje Rysunek 20. [14], [15], [16]



Rysunek 20.a) pierwsza klatka sekwencji obrazów, b) przepływ optyczny, c) uzyskany obraz wyjściowy – druga klatka sekwencji obrazów

Powstało wiele metod wyznaczania przepływu optycznego, które można podzielić na trzy podstawowe grupy:

- metody gradientowe - oparte o analizę pochodnych przestrzennych i czasowych obrazu;
- metody częstotliwościowe – oparte o filtrację częstotliwościową obrazu;
- metody korelacyjne – oparte o dopasowanie odpowiednich fragmentów obrazu;

Każda z tych metod sprawdza się lepiej w określonych zastosowaniach. Najczęściej stosowaną i dającą najlepsze rezultaty jest metoda gradientowa, lecz jej wadą jest jej długi czas wykonywania. Czasami nie jest potrzebne dokładne wyznaczenie przepływu optycznego natomiast istotny jest czas obliczeń, co jest atutem metod częstotliwościowych. Metody korelacyjne są rzadko stosowane do śledzenia obiektów, lecz znalazły zastosowanie do kompresji strumienia video np. w formacie MPEG. [17]

Przepływ optyczny jest powszechnie znaną metodą, lecz możliwości jej zastosowania są dosyć ograniczone. Jest to spowodowane dużym wpływem oświetlenia materiału, z jakiego jest zrobiony śledzony obiekt czy jego przysłonięcia.

Poniżej zamieszczam przykład działania metody przepływu optycznego, na którym ruch obiektu oznaczony jest niebieskimi smugami.

Rysunek 21. Przykład przepływu optycznego

Najczęściej spotykanymi algorytmami wykorzystującymi przepływ optyczny są: [1] ,[14]

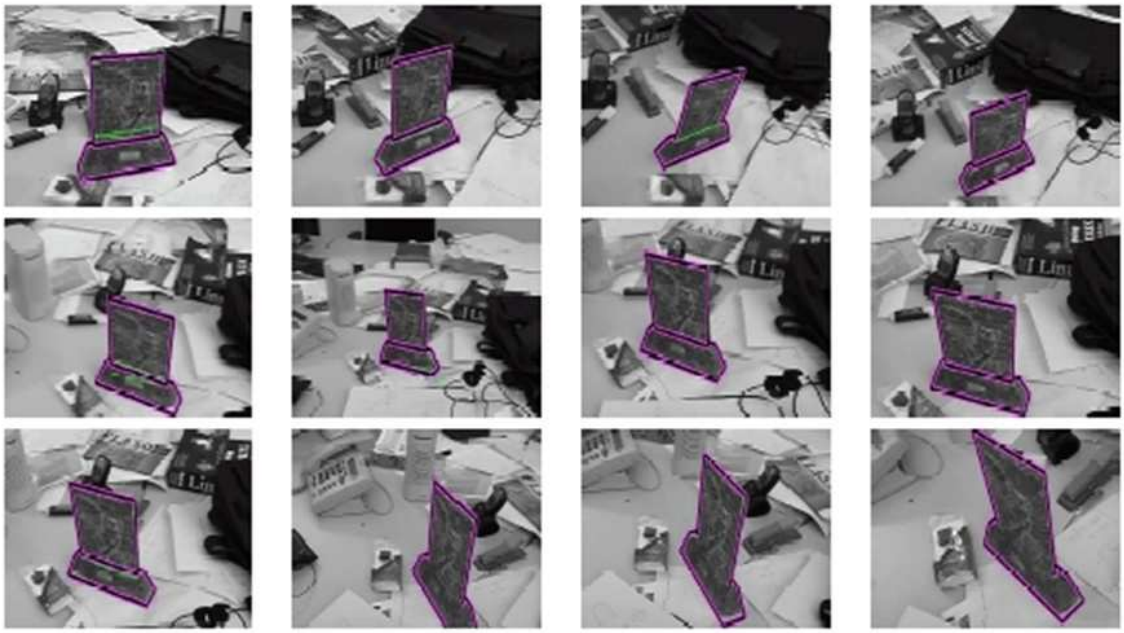
- algorytm Lukas-Kanade – w którym sprawdzane są przemieszczenia poszczególnych pikseli obrazu; metoda ta sprawdza się z niewielkimi przesunięciami, jej założeniem jest niezmienność jasności pikseli oraz ograniczenie do niewielkiego rozmiaru okna;
- algorytm Horn-Schunk – w której wektor przesunięcia wyznaczany jest na podstawie przemieszczenia wybranych obszarów obrazu;

3.3 Śledzenie konturów aktywnych (ang. Active Contour–Based Tracking)

Podstawą tej metody jest uzyskanie obrysu zewnętrznego śledzonego obiektu. W pierwszej kolejności tworzony jest obraz konturowy badanego elementu, a następnie eliminowane są jego kontury wewnętrznych. Aby wynik tych operacji był optymalny istotne jest właściwe przygotowanie obrazu tzn. odfiltrowanie, wygładzenie oraz wzmocnienie kontrastu. Najczęściej do tego celu wykorzystywany jest filtr medianowy.

Po wyznaczeniu konturu zewnętrznego śledzonego obiektu zapisywane są wartości pikseli na jego drodze. W kolejnych klatkach sekwencji obrazów sprawdzane jest ich przesunięcie, które daje obraz ruchu całego obiektu. Ze względu na to iż obiekt zmienia kształt, położenie względem obserwatora, orientację potrzebne jest aktualizowanie obrazu konturowego. [13], [18]

Do wyznaczania krawędzi stosowane są filtry wykrywające kontury, gradientowe czy detektor Harrisa (znajduje miejsca, w których zdecydowanie zmienia się intensywność światła).



Rysunek 22. Przykład działania metody śledzenia konturów

Metoda ta jest skuteczna dla obrazów, dla których obiekt w sposób znaczący różni się od tła. Wynika to z tego, że w takich przypadkach otrzymany kontur zewnętrzny śledzonego obiektu jest dokładniejszy.

3.4 Śledzenie obiektów na podstawie ich cech (ang. Feature-Based Tracking)

Metody zaliczane do tej grupy oparte są na cechach obiektu. W pierwszym etapie tworzony jest wektor cech obiektu, a następnie w kolejnych krokach porównywany jest on z cechami fragmentów z kolejnych klatek badanej sekwencji obrazów. Uzyskanie założonego wyniku porównania oznacza lokalizację obiektu. [13]

Do metod śledzenia opartych o cechy obiektu zaliczane są:

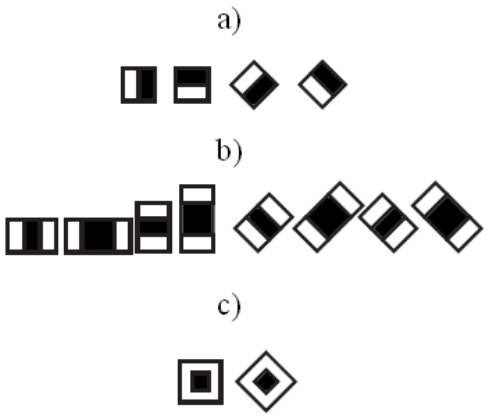
- metoda wykorzystująca cechy Haara (ang. Haar-Like Features);
- metoda wykorzystująca dobre do śledzenia cechy (ang. Good Features to Track);
- metoda wykorzystująca cechy SIFT (ang. Scale Invariant Features Transform);
- metoda wykorzystująca cechy SURF (ang. Speeded Up Robust Features);

3.4.1 Metoda wykorzystująca cechy Haara

Podstawą cech Haara jest stworzona w 1909 r. przez Alfreda Haara falka (ang. Haar Wavelet). Charakteryzuje się ona tym, że odpowiadający jej filtr nie wprowadza przesunięć pomiędzy sygnałem wejściowym, a wyjściowy. Umożliwia to ich kaskadowe łączenie bez potrzeby kompensacji przesunięcia fazy sygnału. Podstawowa falka Haara ma postać: [1] , [19]

$$f(t) = \begin{cases} +1, gdy\ 0 < t < 1/2 \\ -1, gdy \frac{1}{2} \leq\ t < 1 \\ 0, gdy\ t < 0\ lub\ \geq 1 \end{cases}$$

Falka Haara znalazła zastosowanie w procesie śledzenia obiektów przy wyznaczaniu cech obiektu. Klasyfikator Haara przydziela poszczególne piksele do białych i czarnych prostokątów. Określają one czy dany fragment obrazu jest krawędzią, linią czy punktem centralnym. Na Rysunku 23 przedstawione zostały przykładowe cechy wykorzystywane do opisu obiektu:



Rysunek 23.a) cechy krawędzi, b) cechy linii, c) cechy otoczenia centrum

Następnie dla każdej cechy obliczana jest jej wartość, jako różnica pikseli przydzielonych do obszaru czarnego i białego.

$$v_n = \sum p_b(x,y) - \sum p_w(x,y)$$

gdzie:

v_n – wartość cechy;

p_b – wartość piksela przydzielonego do czarnego prostokąta;

p_w – wartość piksela przydzielonego do białego prostokąta;

Na podstawie otrzymanych wartości cech oraz porównaniu ich z przyjętym progiem określany jest jej znak. Na jego podstawie wskazywane jest czy dana cecha dobrze opisuje obiekt (+1) czy nie (-1).

$$f_n = \begin{cases} +1, & gdy\ v_n \geq t_n \\ -1, & gdy\ v_n < t_n \end{cases}$$

gdzie:

f_n – cecha;

v_n – wartość cechy;

t_n – prób;

Z powyższej analizy uzyskiwany jest podstawowy zbiór cech, lecz często jest on niewystarczający. Jest to spowodowane takimi czynnikami jak zmienność kształtu, czy oświetlenia. Otrzymanie optymalnego opisu obiektu jest możliwe dzięki stworzeniu klasyfikatora opartego na bardzo dużej ilości obrazów a co za tym idzie wzorów jednego badanego modelu. Do tego celu został stworzony klasyfikator Viola-Jonesa (ang. Viola-Jones detector). Często bazuje on na tzw. obrazach całkowych (ang. Integral Image),w których wartości pikseli obliczane są z sumy punktów znajdujących się w prostokącie ponad na lewo od badanego punktu. Zależność tę opisuje wzór:

$$O_2(x,y) = \sum_{x' \le x, y' \le y} O_1(x',y')$$

gdzie:

O_1 – obraz wejściowy;

O_2 – obraz całkowy;

Działanie klasyfikatora Viola-Jonesa opiera się na wyznaczaniu cech dla kolejnych obrazów wzorcowych, a następnie ocenie prawdopodobieństwa ich występowania [20]. Jest to realizowane poprzez obliczenie funkcji wagowej cech:

$$f_w = (w_1 f_1 + w_2 f_2 + \cdots + w_i f_i)$$

gdzie:

w_i – prawdopodobieństwo występowania danej cechy (waga);

Wagi są aktualizowane z każdym badanym obrazem, a postać f_w w kolejnych krokach wyliczana jest z sumy wcześniejszych wyników. Na ogół w końcowej postaci brana pod uwagę jest jedynie pierwsza cecha, której waga jest największa, czyli f_1. W taki sposób znajdowana jest cecha, która najlepiej opisuje obiekt badań.

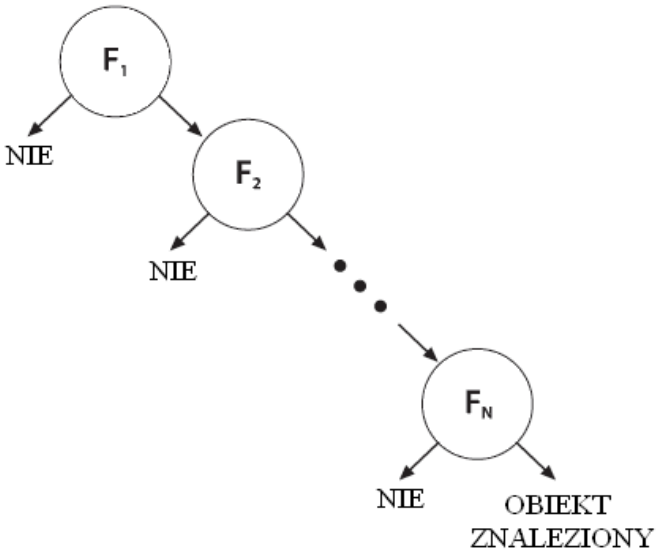
Gdy obiekt zostanie już scharakteryzowany odpowiednimi cechami, kolejne klatki sekwencji obrazów sprawdzane są w poszukiwaniu obszarów zbliżonych opisem do wzorca. Jeżeli występuję zgodność otrzymuje on wartość +1, jeżeli nie -1, a jeżeli trudno ocenić to 0:

$$F = \begin{cases} -1, gdy\ f_w < 0 \\ 0, gdy\ f_w = 0 \\ +1, gdy\ f_w > 0 \end{cases}$$

gdzie:

w_i – waga cechy;

Obiekt zostaje znaleziony, jeżeli kilka leżących obok siebie obszarów jest do niego zbliżonych. Jest to typowa forma klasyfikatora kaskadowego, jakim jest klasyfikator Viola-Jonesa:



Rysunek 24. Schemat działania klasyfikatora Viola-Jonesa

W wypadku tej metody najbardziej istotną częścią jest stworzenie dobrego klasyfikatora, co jest dosyć czasochłonne szczególnie dla nowo badanych obiektów. Metoda ta znalazła zastosowanie w śledzeniu części ludzkiego ciała takich jak twarz, ręce, oczy itd.

3.4.2 Metoda wykorzystująca „cechy dobre do śledzenia”

W metodzie tej wybór cech obiektu dokonywany jest na podstawie dwóch obrazów przedstawiających jego minimalne przesunięcie. W oparciu o nie tworzony jest obraz jego ruchu: [21]

$$O_2(x, y, t + \Delta t) = O_1(x - \xi(x, y, t, \Delta t), y - \eta(x, y, t, \Delta t))$$

gdzie:

t – czas, po którym został wykonany pierwszy obraz;

t + Δt – czas, po którym zostało wykonany drugi obraz;

O_1,O_2 – obrazy wzorcowe;

R – obraz ruchu;

Piksele w obrazie wykonanym w chwili t + Δt można otrzymać poprzez przesunięcie odpowiednich pikseli w obrazie O_1. W ten sposób tworzony jest tzw. wektor przeniesienia δ(ξ,η). Przesunięcie to odniesione jest do pewnego punktu

X(x,y), który odpowiada pozycji małego okna zawierającego badany obiekt na pierwszym obrazie. Wektor δ możemy również przedstawić w inny sposób:

$$\delta = \mathrm{DX} + \mathrm{d}$$

gdzie:

D - macierz deformacji;

$$\mathrm{D} = \begin{bmatrix} d_{xx} & d_{xy} \\ d_{yx} & d_{yy} \end{bmatrix}$$

d- przesunięcie centrum okna;

Dzięki wektorowi przeniesienia można określić pozycję obiektu na kolejnym obrazie znając jego wcześniejsze położenie. Ilustruje to poniższe równanie:

$$O_2(AX + d) = O_1(X)$$

gdzie:

A = 1 + D – 1 jest macierzą jednostkową 2x2;

Często duży wpływ na prawdziwość przeprowadzanych obliczeń mają różnego rodzaju szumy. Ich zmniejszenie uzyskuje się poprzez zastosowanie linearyzacji Taylora, czyli dodanie odpowiedniego współczynnika.

$$O_2(AX + d) = O_2(X) + g^T(\mathrm{u})$$

gdzie:

T- odpowiednio dobrana macierz linearyzująca;

Po przeprowadzeniu powyższych operacji otrzymywane są odpowiednio dobrane cechy. Mogą być nimi np.: narożniki, punktu typu impulsowego, różnego rodzaju obszar oraz wszystkie inne, które w sposób unikatowy opisują obiekt [21]. Uzyskane w ten sposób cechy nie są jednak idealne, gdyż wśród nich mogą występować cechy opisujące duże powierzchnie, które w trakcie śledzenia ulegają największym zmianom. Ich ilość zawężana jest już podczas śledzenia obiektu.

Dla kolejnych klatek sekwencji obrazów dokonywane jest porównanie ich fragmentów ze wzorcem. Następnie na podstawie ilości zgodnych i błędnych cech

dokonywane jest rozpoznanie obiektu. Ze względu na to, że model jest aktualizowany istotne jest, aby cechy, które uległy znacznym zmianom w stosunku do pierwotnych były likwidowane. Dzięki temu wykorzystywane są tylko te własności obiektu, które nadają się do śledzenia, a eliminuje te, które wprowadzają błędy.

3.4.3 Metoda wykorzystująca cechy SIFT

Charakterystyczne dla tej metody jest to, że uzyskane cechy obiektu są niezależne od skali oraz orientacji (ang. Scale Invariant Feature Transform). Taki opis wzorca sprawia, że jest on bardziej indywidualny, co zwiększa prawdopodobieństwa jego wykrycia [22]. Ekstrakcja cech dokonywana jest poprzez następujące czynności:

- znalezienie fragmentów obrazu niezmiennych w stosunku do skali oraz orientacji;
- wybór na ich obszarze punktów bazowych, które są najbardziej niezmienne;
- ocena orientacji punktów bazowych na podstawie gradientów obszarów;
- opis zmian punktów bazowych, jego kształtu, jasności, miejsca;

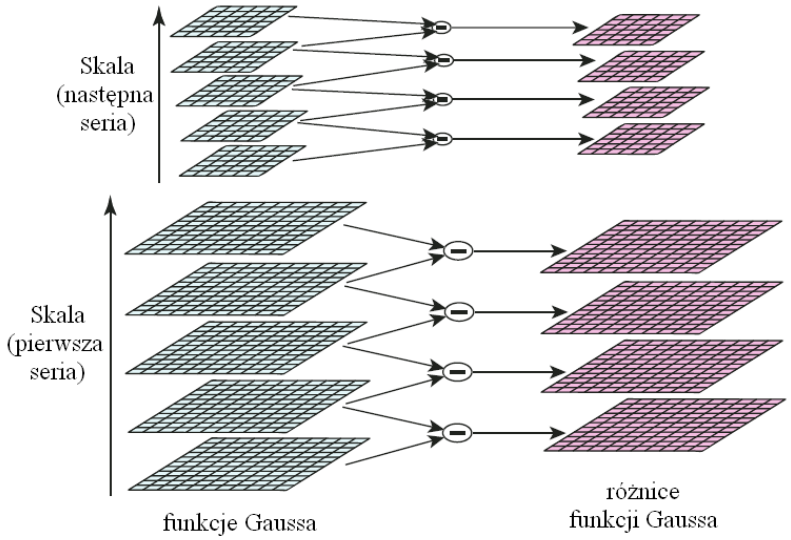
Odpowiedni dobór obszarów jest uzyskiwany poprzez zastosowanie piramidy Gaussa [23], w której każdy z jej poziomów różni się od siebie odpowiednią skalą. Wartości pikseli dla poszczególnych etapów obliczane są z zależności:

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

gdzie:

σ – współczynnik Gaussa; np. σ =$\sqrt{2}$

Dla każdej przyjętej skali wyznaczane są kolejne obrazy o różnych współczynnikach σ, a następnie obliczane są ich różnice. Ogólną tego zasadę przedstawia poniższy rysunek:



Rysunek 25. Schemat wyznaczania obszarów – piramida Gaussa

Różnica pomiędzy poszczególnymi obrazami wyliczana jest ze wzoru:

$$D(x, y, \sigma) = \big(G(x, y, k\sigma) - G(x, y, \sigma)\big) * O(x, y)$$

gdzie:
O – obraz wzorcowy;
k- współczynnik skali;

Na podstawie powyższych operacji wyszczególniane są fragmenty obrazu (obiektu), z których będą wybierane cechy. Pierwszą wykonywaną czynnością jest wyznaczenie lokalnych maksimów i minimów. Następnie z pośród nich wybierane są te piksele, których wartość jest największa lub najmniejsza w obszarze z 8-ma sąsiadami na obrazie wzorcowym oraz 9-oma dla zwiększonej i zmniejszonej skali. Powyższe obliczenia dają obraz wygładzony.

$$L(x, y) = G(x, y, \sigma) * O(x, y)$$

gdzie:
L – obraz wygładzony;

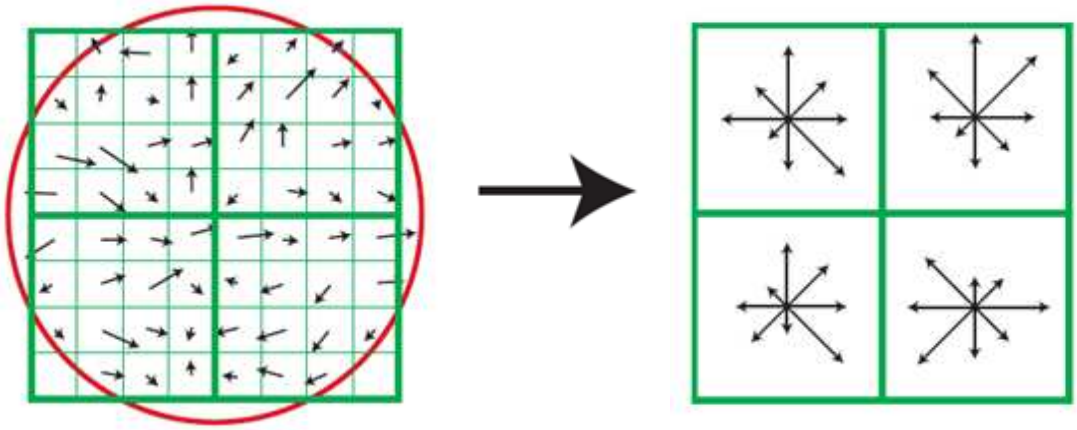
Następnie na jego podstawie wyznaczane są gradienty oraz orientacje poszczególnych punktów bazowych:

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

$$\theta(x,y) = \tan^{-1}((L(x,y+1) - L(x,y-1))/(L(x+1,y) - L(x-1,y)))$$

gdzie:
m – gradient;
θ – orientacja;

a) b)



Rysunek 26. a) obraz gradientów, b) orientacje punktów bazowych

Uzyskane punkty bazowe, ich orientacja, obserwacja zmian w stosunku do skali stanowią cechy obiektu. Odnalezienie badanego obiektu w kolejnych klatkach sekwencji obrazów odbywa się poprzez porównanie ich fragmentów z wzorcem. Nie są jednak sprawdzane całe obszary, lecz tylko punkty bazowe, ich wartości, orientacja oraz przesunięcie względem poprzedniego obrazu, którym w pierwszej kolejności jest wzór. Jeżeli są one zbliżone obiekt jest rozpoznawany i oznaczany.

Metoda ta szczególnie dobrze sprawdza się dla obiektów bardziej skomplikowanych, dla których możliwe jest wyznaczenie jak największej ilości cech. Takimi obiektami mogą być np. twarze, książki, samochody itd.

3.4.4 Metoda wykorzystująca cechy SURF

Zasada działania tej metody jest w dużym stopniu zbliżona do opisanej powyżej. Kolejność wykonywanych operacji jest taka sama, lecz ich podstawową różnicą jest sposób detekcji obszarów oraz ekstrakcji cech. W metodzie cech SURF do tych celów wykorzystywana jest macierz Hessa (ang. Hessian matrix) postaci: [24]

$$H(x,\sigma) = \begin{bmatrix} L_{xx}(X,\sigma) & L_{xy}(X,\sigma) \\ L_{xy}(X,\sigma) & L_{yy}(X,\sigma) \end{bmatrix}$$

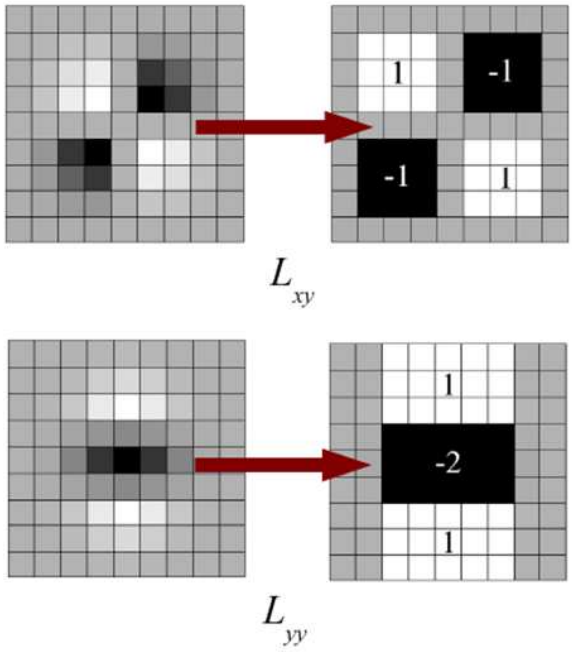
gdzie:

X – punkt (x,y) badanego obrazu;

σ – współczynnik skali;

L_{xx} – przybliżenie Gaussa $\frac{\partial^2}{\partial x^2} g(\sigma)$ itd.

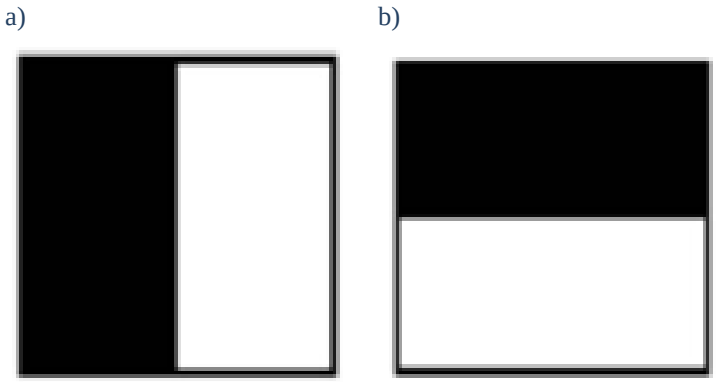
Zastosowanie macierzy Hessa przyśpiesza detekcję oraz zwiększa dokładność tej operacji. Wyznaczenie przybliżeń Gaussa jest dosyć czasochłonne, dlatego wprowadzono do obliczeń obrazy całkowe. Poniższe rysunki dobrze ilustrują zasadę ich powstawania. Istotne jest również wcześniejsze odfiltrowanie obrazu.



Rysunek 27. Przykład wyznaczania L_{xy} i L_{yy}

Poprzez analizę zmian wybranych obszarów dla różnych skal oceniane jest, które fragmenty najbardziej nadają się do śledzenia. Do tego celu wykorzystywana jest piramida Gaussa, tak samo jak dla metody cech SIFT. W ten sposób otrzymane są obszary charakterystyczne dla danego, danych obiektów.

Kolejnym ważnym etapem jest wyznaczenie orientacji wybranych wcześniej obszarów, co jest realizowane przy użyciu falki Haara. Dla określonych fragmentów obrazu sprawdzana jest przynależność poszczególnych pikseli do białych i czarnych obszarów. Orientacja określana jest na podstawie uzyskanych cech Haara.



Rysunek 28. Przykład cech wykorzystywanych do wyznaczenia orientacji, a) pozioma, b) pionowa

Rezultatem końcowym są zorientowane cech, na których podstawie rozpoznawany jest obiekt w kolejnych klatkach sekwencji obrazów. W procesie śledzenia istotne są tylko te fragmenty, których kontrast jest zbliżony do wzorca. Aby ograniczyć przeszukiwany obszar wyznaczany jest on w oparciu o ruch obiektu, a mianowicie sumy $L_{xx}+ L_{yy}$ macierzy Hessa..

Metoda ta sprawdza się dla bardziej skomplikowanych obiektów tzn. takich, które mają dużo cech charakterystycznych. Wykazuje lepszą dokładność dla elementów obracających się czy rozmytych oraz jest dużo szybsza niż metoda wykorzystująca cechy SIFT. Działa jednak gorzej przy słabszym oświetleniu.

3.5 Śledzenie na podstawie modelu obiektu (ang. Model-Based Tracking)

Jest to najbardziej rozpowszechniony sposób śledzenia obiektów. Opiera się tworzeniu modelu obiektu, którym w najprostszym rozumieniu może być zdjęcie badanego obiektu. W kolejnych klatkach sekwencji obrazów poszczególne piksele porównywane są z wzorcem i na tego podstawie element jest rozpoznawany. Drugim również bardzo popularnym i uniwersalnym modelem jest rozkład kolorów badanego obiektu. Uzyskuje się go poprzez wyznaczenie histogramu. Bardzo istotne dla obu rozwiązań jest ustalenie progu dopasowania określającego, kiedy wybrany fragment obrazu zawiera obiekt [13]. Do metod wykorzystujących powyższe mechanizmy zaliczamy:

- metoda dopasowania do szablonu (ang. Template Matching Tracking);
- metoda śledząca przesunięcie „jądra" obiektu (ang. Kernel-Based Tracking);
- metoda ciągłej adaptacji przesunięcia obiektu (ang. Continuously Adaptive MeanShift);

3.5.1 Metoda dopasowania do szablonu

W metodzie tej modelem obiektu jest jego zdjęcie. W kolejnych klatkach sekwencji obrazów ich fragmenty porównywane są z wcześniej otrzymanym wzorem. Zgodność odpowiedniej liczby pikseli klasyfikuje nam czy jest to poszukiwany obiekt czy nie [1]. Najczęściej spotykane są trzy metody porównawcze:

- metoda różnicy kwadratów

$$R(x,y) = \sum_{x'y'} [W(x',y') - O(x+x',y+y')]^2$$

gdzie:

W - obraz wzorcowy;

O - sprawdzany obraz;

Uzyskanie sumy kwadratów bliskiej 0 oznacza znalezienie obiektu, natomiast w przeciwnym wypadku taki obraz jest odrzucany.

- metoda korelacji

$$R(x,y) = \sum_{x'y'} [W(x',y') \cdot O(x+x', y+y')]^2$$

Metoda ta jest przeciwieństwem poprzedniej gdyż dla niej uzyskanie dużej wartości sumy kwadratów oznaczania znalezienie badanego obiektu.

- metoda korelacji koefektywnej

$$R(x,y) = \sum_{x'y'} [W'(x',y') \cdot O'(x+x', y+y')]^2$$

gdzie:
W’ - obraz względny do obrazu wzorcowego;
O’ – obraz względny do obrazu sprawdzanego;

$$W'(x',y') = W(x,y) - \frac{1}{(w*h)\sum_{x",y"} W(x,\mathrm{y})}$$

gdzie:
w – długość obrazu;
h – wysokość obrazu;

$$O'(x',y') = I(x+x', y+y') - \frac{1}{(w*h)\sum_{x",y"} O(x+x",\mathrm{y}+\mathrm{y}")}$$

Metoda zbliżona do metody korelacji z tą jedyną różnicą, że dopasowanie obrazu z wzorcem występuje dla sumy kwadratów równej 1.

Trzy powyższe metody są dosyć mało odporne na zmiany oświetlenia, co powoduje błędy podczas wykrywania obiektu. Problem ten został rozwiązany przez Galtona, który wprowadził czynnik normalizujący postaci: [1]

$$N(x,y) = \sqrt{\sum_{x'y'} W(x',y')^2 \cdot \sum_{x'y'} O(x+x',y+y')^2}$$

Nowe wartości sumy kwadratów wykorzystywane w powyższych metodach otrzymywane są poprzez podzielenie podstawowych przez czynnik normalizujący.

Metoda dopasowania szablonu sprawdza się, gdy obiekt nie ulega zbyt dużym zmianom podczas procesu śledzenia. Częściej jednak wykorzystuje się ją do sprawdzania oryginalności różnych elementów graficznych takich jak np. banknoty.

3.5.2 Metoda śledząca przesunięcie „jądra" obiektu

Metoda ta jest powszechnie nazywana, jako „MeanShift". Została stworzona do wyznaczania maksimum lokalnej funkcji gęstości prawdopodobieństwa (ang. probabilisty den sity function), lecz wraz z szybkim rozwojem techniki znalazła zastosowanie w innych dziedzinach takich jak np. przetwarzanie obrazów. Jej działanie polega na znalezieniu tzw. jądra (ang. Kernel), czyli fragmentu o najgęstszym rozkładzie danej cechy. Uzyskuje się to poprzez coraz to bliższe dopasowanie funkcji rozkładu jej prawdopodobieństwa do badanego elementu. Poniżej przybliżę działanie tej metody w zastosowaniu do śledzenia obiektów w sekwencjach obrazów.

Pierwszym etapem jest wyznaczenie funkcji rozkładu. Powszechnie do tego celu wykorzystywany jest histogram interesującego nas fragmentu. Wiedząc, że poszczególne wartości pikseli mogą występować z określonym prawdopodobieństwem, można wyznaczyć ich rozkład. [25], [26]

$$\vec{q} = \{q_i\}_{i=1\ldots m} \text{ dla modelu}$$

oraz

$$\vec{p}(y) = \{p_i(y)\}_{i=1\ldots m} \text{ dla „kandydata"}$$

gdzie:

m – ilość przedziałów wartości cech;

y – punkt startowy;

W tym wypadku, jako model przyjmowany jest obraz obiektu wzorcowego natomiast za „kandydata" obraz, który będzie z nim porównywany. Na podstawie ich rozkładów powstaje funkcja, przy pomocy której badany element będzie rozpoznawany.

$$f(y) = f[\vec{q}, \vec{p}(y)]$$

Wartości funkcji rozkładu dla poszczególnych cech, przy wykorzystaniu estymatora gęstości jądra (ang. Kernel Density Estimation) obliczane są z poniższych wzorów:

$$q_i = C \sum_{i=1\ldots m} K\left(\left|\left|x_j\right|\right|^2\right) \text{ dla modelu}$$

gdzie:

C– stała normalizująca, często przyjmowana, jako$\frac{1}{n}$ gdzie n jest liczbą pikseli w „jądrze";

x_j– pozycja wybranego piksela;

K – postać „jądra";

$$p_i(y) = C_h \sum_{i=1\ldots m} K\left(\left|\left|\frac{y_0 - x_j}{h}\right|\right|^2\right) \text{dla „kandydata"}$$

gdzie:

C_h – stała normalizująca, często przyjmowana, jako$\frac{1}{nh}$ gdzie n jest liczbą pikseli w „jądrze";

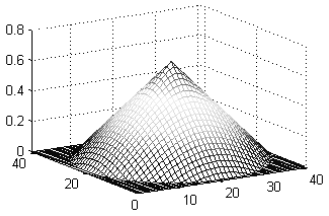
h – promień okna;

y_0 – pozycja początkowa;

W literaturze najczęściej występują dwa rodzaje „jądra”:

- *jądro Epanecznikowa*

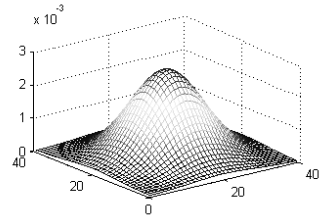
$$K_E = \begin{cases} \frac{1}{2} n^{-1}(d+2)(1-||x||^2) & gdy\ ||x|| < 1 \\ 0 & gdy\ ||x|| \geq 1 \end{cases}$$



gdzie:
n – liczba pikseli jądrze;
d – liczba przestrzeni;

- *jądro Gaussa*

$$K_N(x) = (2\Pi)^{-d/2} e^{(-||x||^2/2)}$$



Dzięki uzyskaniu powyższych informacji określić początkową funkcję przystosowania. Wyznaczamy ją w oparciu o współczynnik Bhattacharya, czyli sumę pierwiastków iloczynu funkcji rozkładu.

$$f(y) = f[\vec{q}, \vec{p}(y)] = \sum_{i=1}^{m} \sqrt{p_i(y) q_i}$$

W tym kroku otrzymywana jest wartość funkcji rozkładu dla punku początkowego y = y_0. Będzie ona podstawą obliczeń przybliżających algorytm do uzyskania maksymalnego dopasowania - kandydata do wzorca. Aby ocenić czy aktualna pozycja jest bliska „centrum” konieczne jest wyznaczenie nowego punktu badań. Otrzymywany jest on poprzez obliczenie wektora przesunięcia:

$$M_h(y) = \left[\frac{\sum_{i=0}^{n} w_i(y)x_i}{\sum_{i=0}^{n} w_i(y)}\right] - y]$$

gdzie:

n – liczba pikseli w jądrze;

x_i– pozycja wybranego piksela;

y – pozycja (w tym kroku $y=y_0$)

w_i – waga dobrana rodzaju jądra;

Po otrzymaniu wektora przesunięcia jest on dodawany do pozycji początkowej, dzięki czemu uzyskiwany jest kolejny punkt badań y_1. Od tego momentu wszystkie operacje wykonywane są od początku wedle powyższego opisu, aż do uzyskania wartości wektora M_h mniejszej od przyjętego progu dokładności.

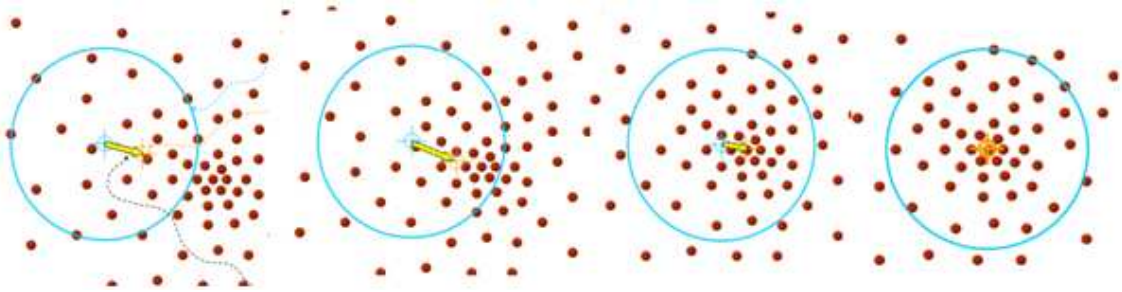
W sytuacji, gdy po przesunięciu punktu y w nowe miejsce zmniejszy się wartość funkcji rozkładu następną jego pozycję określa wzór:

$$y_s = \frac{y_{s-1} + y_{s-2}}{2}$$

Podsumowując - zasada działania algorytmu „Mean-Shift” polega na:

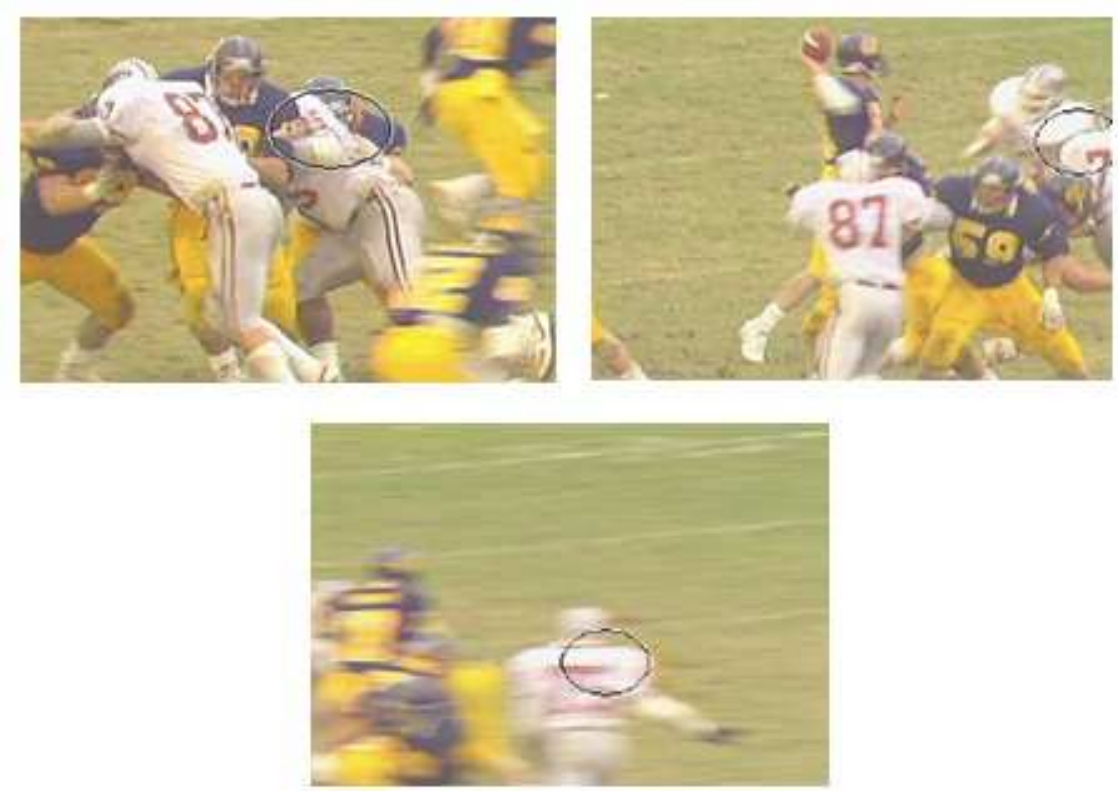
1) Obliczeniu funkcji rozkładu dla wzorca oraz „kandydata”
2) Wyznaczeniu funkcji dopasowania obu elementów
3) Wyznaczeniu wektora przesunięcia
4) Porównaniu jego wartości z progiem dokładności

Poniżej zamieszczony został schemat oraz przykład jego działania.



Rysunek 29. Schemat działania algorytmu Mean-Shift

Przykład:



Rysunek 30. Śledzenie zawodnika w kolejnych klatkach sekwencji obrazów przy użyciu algorytmu „Mean-Shift” [25]

Ze względu na wykorzystanie w tej metodzie operacji histogramu sprawdza się, gdy badany obiekt jest zdecydowanie różny od tła, jego kolorystyka jest unikatowa. Częściej jednak omawiana metoda występuje jako bardziej złożonych algorytmów np. Camshift (ang. Continously Adaptive Mean-Shift). W przetwarzaniu obrazów stosowana jest nie tylko do śledzenia obiektów, ale również filtracji czy segmentacji obrazów, oddzielania modelu od tła, aproksymacji gęstości kolorów itd.

3.5.3 Metoda ciągłej adaptacji przesunięcia obiektu

Metoda ta powszechnie nazywana „Camshift” oparta jest o powyżej opisany algorytm Mean-Shift i składa się z następujących etapów:

- tworzenie histogramu w oparciu o obraz badanego obiektu;
- wyznaczenie funkcji prawdopodobieństwa, która otrzymywana jest np. z rzutu składowej H na płaszczyznę obrazu dla modelu HSV. Przy jej pomocy wyznaczany jest obraz, na którym pozostawione są jedynie te piksele, dla których wartość funkcji prawdopodobieństwa jest odpowiednio duża;
- lokalizacja punktu centralnego obiektu (jego środka) za pomocą algorytmu Mean-Shift na wcześniej otrzymanym obrazie;
- określenie wymiarów obiektu na podstawie parametrów okna, w którym się mieści;

Uzyskane powyżej parametry badanego obiektu wykorzystywane są do określenia pozycji początkowej algorytmu w następnej klatce sekwencji obrazu. [27], [28] Metoda ta jest rozszerzeniem algorytmu Mean-Shift o dodatkowe wyznaczenie wymiarów obiektu. Jest to bardzo przydatne, gdy mamy do czynienia z obiektami o podobnych histogramach, lecz różniące się kształtem. Popularnym zastosowaniem algorytmu CAMSHIFT jest śledzenie ludzi, twarzy, rąk itp.

3.6 Metody hybrydowe

W powyższych rozdziałach opisane zostały podstawowe metody śledzenia obiektów w sekwencjach obrazów. Od momentu ich stworzenia powstało wiele ich modyfikacji, które owocują nowymi metodami opartymi na wcześniej odkrytych mechanizmach. Często można również spotkać się z algorytmami łączącymi kilka metod w jedną, udoskonalając w ten sposób ich działanie dla odpowiednich zastosowań. Do metod hybrydowych zaliczane są m.in.:

- śledzenie regionów, szablonu czy cech obiektu z wykorzystaniem algorytmu Mean-Shift; [29], [30]

- śledzenie według przepływu optycznego z wykorzystaniem śledzenia regionów, konturów czy cech obiektu; [31], [32], [33]
- śledzenie dopasowania szablonu opartego na cechach obiektu; [34]

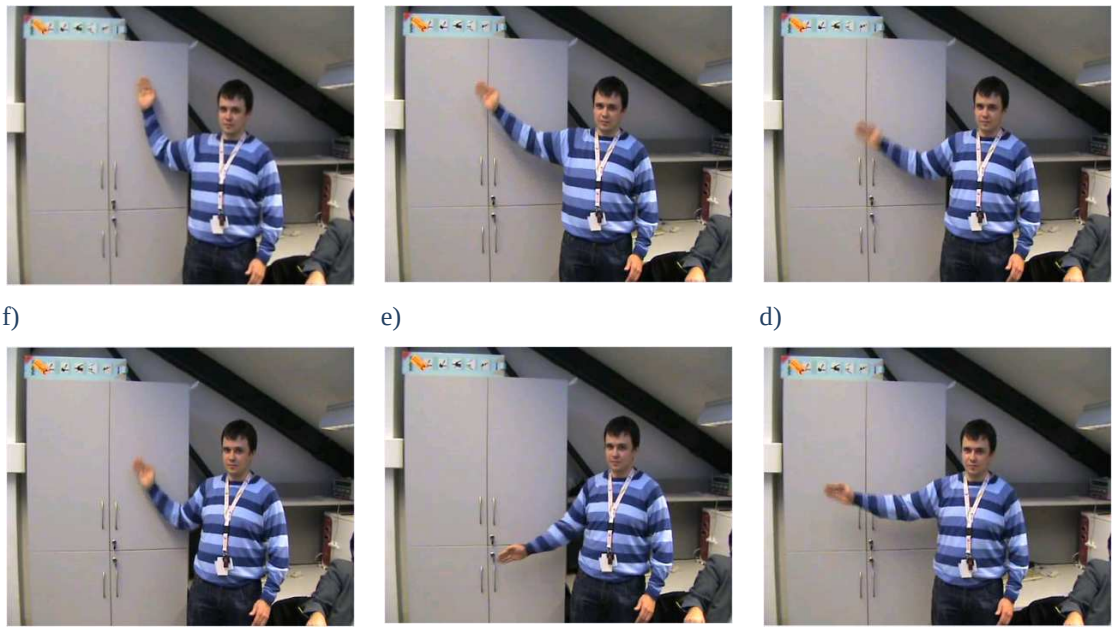
4 Implementacja i analiza porównawcza wybranych algorytmów śledzenia obiektów

4.1 Wprowadzenie

Metod śledzenia obiektów jak i różnych rodzajów ich zastosowań jest wiele. W związku z tym, w poniższym rozdziale przebadane zostały jedynie wybrane, najbardziej popularne metody. Działanie każdej z nich zostało przeanalizowane dla różnych sekwencji obrazów. Zmierzone zostały parametry takie jak: czas przetwarzania i względna skuteczność działania algorytmu (liczba klatek sekwencji, w których obiekt został prawidłowo zlokalizowany w stosunku do pełnej liczby klatek). Badania wykonywane były dla 100 klatek każdej sekwencji.

Do badań wykorzystane zostały poniższe sekwencje obrazów:

a) b) c)



f) e) d)

Rysunek 31. Sekwencja 1

Opis:

W sekwencji 1 przedstawionej na Rysunku 31. badanym obiektem jest prawa dłoń o wymiarach: 34x68 pikseli. Jej kolorystyka jest odmienna od tła, lecz zbliżona do kolorystyki innych obiektów takich jak: lewa dłoń i twarz (ich znaczna odległość od obiektu nie wpływa na proces śledzenia).Obiekt porusza się jednostajnie po trajektorii równoległej do płaszczyzny obrazu (brak efektu zmiany skali) oraz ulega procesowi rotacji.

a) b) c)

d) e) f)

Rysunek 32.Sekwencja 2

Opis:

W sekwencji 2 przedstawionej na Rysunku 32. badanym obiektem jest balon o wymiarach: 80 x 90 pikseli. Jego zróżnicowana kolorystyka zawiera częściowo fragmenty zbliżone do tła, co utrudnia proces jego śledzenia. Obiekt w niewielkim stopniu zmienia swoje położenie między kolejnymi obrazami sekwencji. Nie występują również dodatkowe zaszumienia sekwencji takie jak: słabe warunki zewnętrzne czy jakość działania urządzenia rejestrującego.

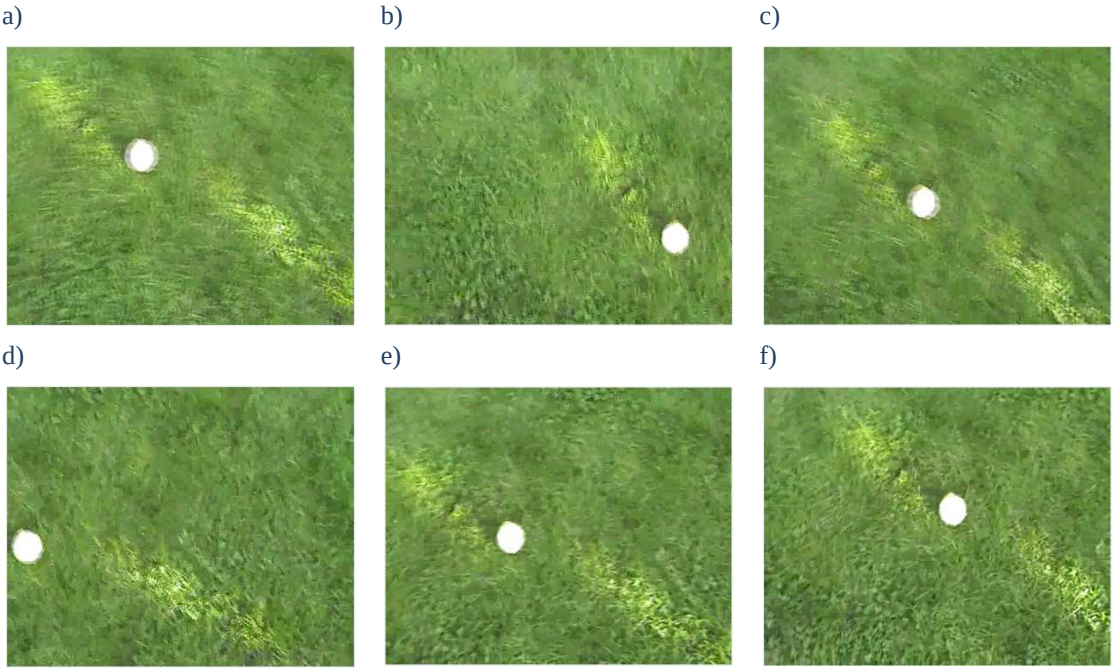
a) b) c)

d) e) f)

Rysunek 33. Sekwencja 3

Opis:

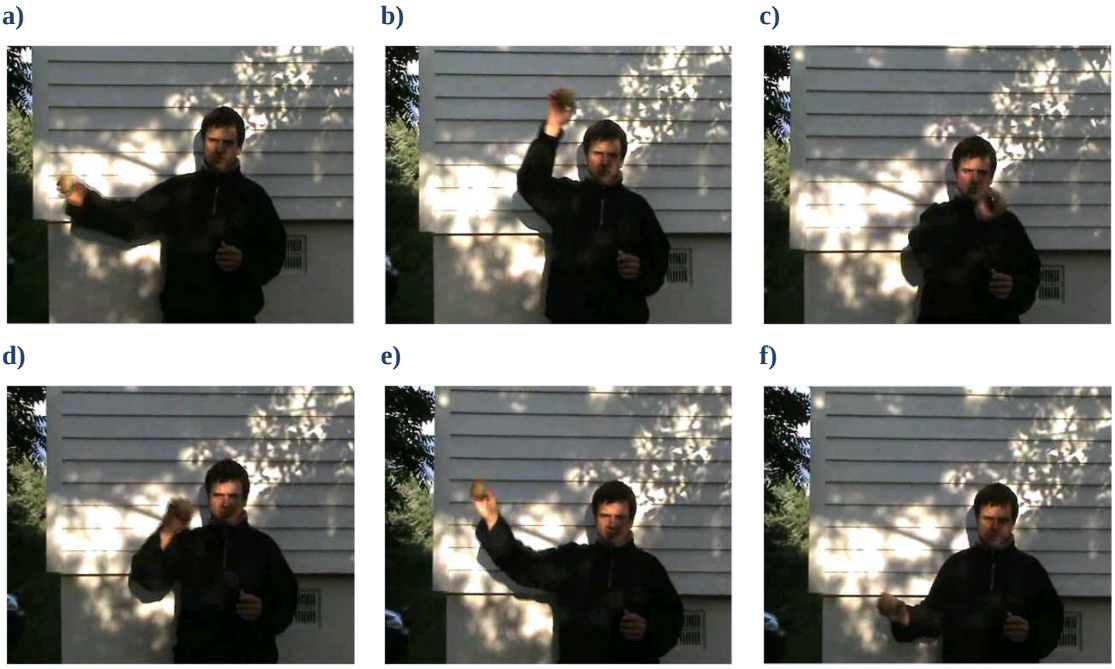
W sekwencji 3 przedstawionej na Rysunku 33. badanym obiektem jest kwiat słonecznika o wymiarach: 70 x 75 pikseli. Jego kolorystyka jest odmienna od tła, lecz bardzo zbliżona do kolorystyki innych obiektów (kwiatów tej samej rośliny znajdującej się w pobliżu badanego obiektu). Ruch obiektu jest zmienny, czego rezultatem są smugi i rozmycia obrazu, co w połączeniu z niedostatecznym oświetleniem znacząco utrudnia proces śledzenia. W tej sekwencji ruch wykonuje urządzenie rejestrujące, a nie badany obiekt.



Rysunek 34. Sekwencja 4

Opis:

W sekwencji 4 przedstawionej na Rysunku 34. badanym obiektem jest piłka tenisowa o wymiarach: 60 x 65 pikseli. Jej kolorystyka jest całkowicie odmienna od tła, a w sekwencji obrazów nie występują żadne inne obiekty. Badany obiekt porusza się płynnie, a jego wymiary względem urządzenia rejestrującego w kolejnych klatkach sekwencji nie ulegają zmianie.



Rysunek 35. Sekwencja 5

Opis:

W sekwencji 5 przedstawionej na Rysunku 35. badanym obiektem jest piłka tenisowa o wymiarach: 30 x 35 pikseli. Jej kolorystyka jest zbliżona do kolorystyki dłoni, w której się znajduje, lecz odmienna od pozostałego tła. Ruch obiektu jest jednostajny w stałej odległości względem urządzenia rejestrującego. Obrazy sekwencji są w znacznym stopniu zaszumione słabym oświetleniem, co może utrudniać proces śledzenia

Rysunek 36. Sekwencja 6

Opis:

W sekwencji 6 przedstawionej na Rysunku 36. badanym obiektem jest prawa ręka o wymiarach: 30 x 60 pikseli. Jej kolorystyka jest odmienna od kolorystyki tła, a na obrazach sekwencji występują inne zbliżone do badanego obiekty. Ruch obiektu jest zmienny w czasie. Dodatkowe utrudnienia w lokalizacji obiektu wprowadza bardzo słabe oświetlenie oraz przesłonięcie przez inny obiekt podczas trwania sekwencji.

a) b) c)

d) e) f)

Rysunek 37. Sekwencja 7

Opis:

W sekwencji 7 przedstawionej na Rysunku 37. badanym obiektem jest piłka tenisowa o wymiarach: 25 x 30 pikseli. Obiekt jak i sekwencja są zbliżone do tych z Rysunku 35., lecz w tym wypadku występują dużo mniejsze zaszumienia wynikające ze słabego oświetlenia, zaś większe spowodowane zmiennym ruchem obiektu.

a) b) c)

d) e) f)

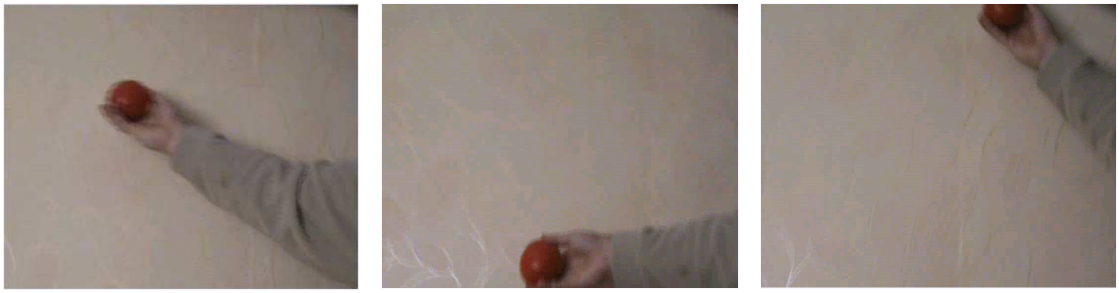
Rysunek 38. Sekwencja 8

Opis:

W sekwencji 8 przedstawionej na Rysunku 38. badanym obiektem jest piłka tenisowa o wymiarach: 40 x 45 pikseli. Obiekt jest nieznacznie zbliżony kolorystycznie do fragmentów tła, a jego kształt i wymiary pozostają stałe w kolejnych klatkach sekwencji. Ruch badanego obiektu jest stały w czasie i nie wpływają na niego dodatkowe zaszumienia.

a) b) c)

d) e) f)



Rysunek 39. Sekwencja 9

Opis:

W sekwencji 9 przedstawionej na Rysunku 39. badanym obiektem jest pomidor o wymiarach 90x90. Obiekt odróżnia się kolorystycznie od tła. Jego ruch jest jednostajny, a podczas trwania sekwencji w niewielkim stopniu zmienia się jego kształt. Dodatkowym utrudnieniem jest fakt, że na niektórych klatkach badany obiekt częściowo wychodzi poza okno rejestracji.

a) b) c)

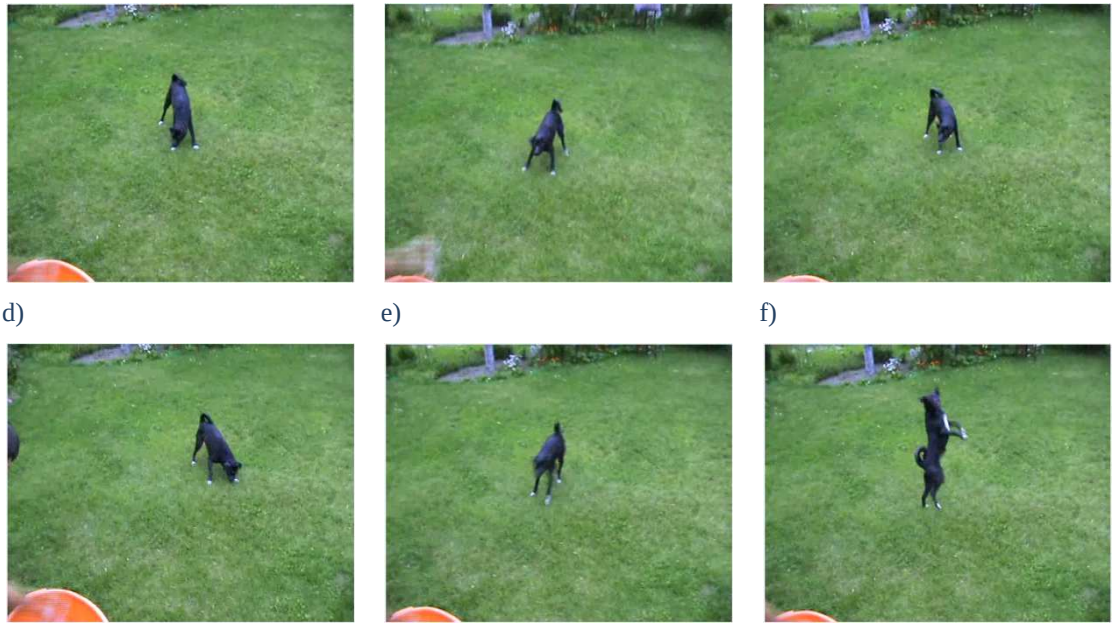
d) e) f)

Rysunek 40. Sekwencja 10

Opis:

W sekwencji 10 przedstawionej na Rysunku 40. badanym obiektem jest piłka tenisowa. Obiekt jest dwukolorowych o kolorystyce odmiennej od kolorystyki tła. Podczas trwania sekwencji ulega on rozmyciu spowodowanemu gwałtownymi zmiany prędkości ruchu (Rysunek 40.b). Dodatkowym utrudnieniem procesu lokalizacji jest występowanie przysłonięć spowodowanych przez inne obiekty znajdujące się w badanej sekwencji (obrazuje to Rysunek 40.e).

a) b) c)



d) e) f)

Rysunek 41. Sekwencja 11

Opis:

W sekwencji 11 przedstawionej na Rysunku 41. badanym obiektem jest czarny pies o wymiarach: 47 x 162 pikseli. Jego kolorystyka jest odmienna od tła oraz innych obiektów znajdujących się w sekwencji. Obiekt nie zmienia znacząco położenia względem urządzenia rejestrującego, lecz ulega procesowi rotacji. Na obrazach sekwencji nie występują dodatkowe zakłócenia.

a) b) c)

d) e) f)

Rysunek 42. Sekwencja 12

Opis:

W sekwencji 12 przedstawionej na Rysunku 42. badanym obiektem jest czerwony samochód o wymiarach: 67 x 126 pikseli. Jego kolorystyka jest odmienna od tła, jak również innych obiektów znajdujących się w analizowanej sekwencji. Ruch obiektu jest jednostajny. Na obrazach sekwencji występują nieznaczne zakłócenia wynikające z niedostatecznego oświetlenia.

a) b) c)

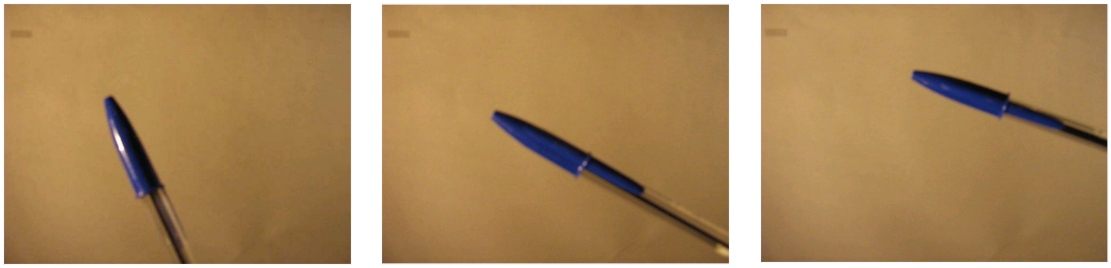
d) e) f)

Rysunek 43. Sekwencja 13

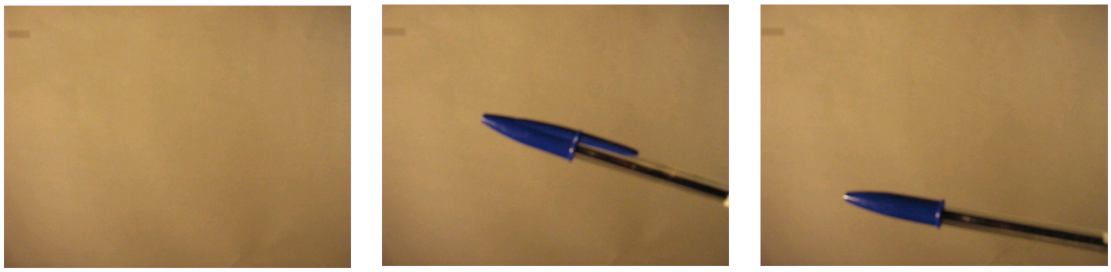
Opis:

W sekwencji 13 przedstawionej na Rysunku 43. badanym obiektem jest czerwony samochód o wymiarach: 50x 72 pikseli. Jego kolorystyka jest odmienna od tła i innych obiektów znajdujących się na analizowanej sekwencji. Zmiana położenia obiektu jest płynna i stała w czasie. Obiekt przybliża się względem urządzenia rejestrującego jednocześnie ulegając procesowi rotacji. Na klatkach sekwencji nie występują dodatkowe zaszumienia.

a) b) c)



d) e) f)

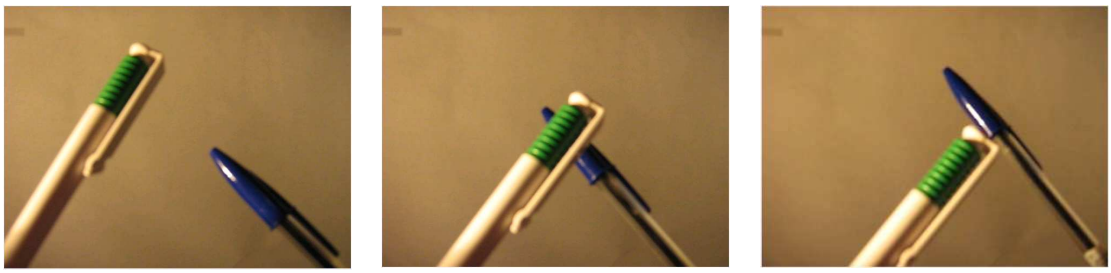


Rysunek 44. Sekwencja 14

Opis:

W sekwencji 14 przedstawionej na Rysunku 44. badanym obiektem jest niebieski długopis wymiarach: 45x 195 pikseli. Jego kolorystyka jest odmienna od tła. Obiekt zlokalizowany jest w stałej odległości względem urządzenia rejestrującego, lecz ulega procesowi rotacji. Dodatkowym utrudnieniem jest fakt, że w trakcie trwania sekwencji obiekt na jakieś czas wychodzi poza okno rejestracji. Na klatkach sekwencji nie występują dodatkowe zaszumienia.

a) b) c)



d) e) f)

Rysunek 45. Sekwencja 15

Opis:

W sekwencji 15 przedstawionej na Rysunku 45. badanym obiektem jest niebieski długopis o wymiarach: 46 x190 pikseli. Jego kolorystyka jest odmienna od tła, jak również drugiego obiektu występującego w analizowanej sekwencji. Ruch obiektu jest jednostajny. W trakcie trwania sekwencji obiekt zostaje przysłonięty przez inny obiekt oraz wychodzi poza okno rejestracji. Na klatkach sekwencji występują dodatkowo zaszumienia wynikające ze słabego oświetlenia.

Aby w jak najlepszy sposób porównać wybrane metody istotne było znalezienie środowiska umożliwiającego stosunkowo prostą implementację wybranych metod śledzenia obiektów oraz umożliwiającego stosunkowo łatwą analizę ich działania. Dlatego do testów wykorzystany został język C z użyciem biblioteki graficznej OpenCV. Dodatkowym jego atutem był fakt, iż środowisko to udostępnia już zaimplementowane funkcje śledzenia obiektów w sekwencjach obrazów.

4.2 Biblioteka OpenCV

OpenCV (ang. Open Source Computer Vision) jest stworzoną przez firmę Intel Corporation biblioteką o otwartym kodzie źródłowym, dedykowaną dla języków C, C++. Można z niej korzystać pod różnymi systemami operacyjnymi takimi jak: Linux, Windows, Mac OS X. Podstawowym jej przeznaczeniem jest przetwarzanie obrazów w czasie rzeczywistym, lecz umożliwia ona również korzystanie i zapisywanie różnego typu plików takich jak np. JPG czy AVI. Oprócz tego wbudowanych jest ponad 500 funkcji z różnych obszarów wizji. Wszystkie dostępne funkcje umieszczone są w pod-bibliotekach: [1]

cv– zawiera funkcje to przetwarzania obrazów, analizy strukturalnej, analizy ruchowej oraz śledzenia ruchu obiektów w obrazach video;

cxcore – zawiera funkcje umożliwiające operacje na plikach, modyfikacje obrazów, dynamiczne zmiany, rysowanie oraz dodawanie elementów do obrazów, kontrola błędów;

highgui –obsługuje operacje wejścia – wyjścia, np. myszka (odczyt z kursora), ekran (wyświetlanie);

cvcam – uniwersalny moduł do współpracy z urządzeniami zewnętrznymi takimi jak kamera czy aparat;

Biblioteka ta zawiera wiele przydatnych funkcji umożliwiających wykonywania wielu operacji na obrazach oraz ich sekwencjach takich jak np.:

- tworzenie oraz udostępnianie
- operacje arytmetyczne oraz logiczne
- filtracja
- segmentacja
- transformata
- operacje morfologiczne
- zmiana palety barw
- histogram
- transformacje geometryczne
- śledzenie obiektów
- rozpoznawanie gestów

Oraz wiele innych również bardziej skomplikowane.

Umożliwia ona również współpracę z urządzeniami zewnętrznymi takimi jak aparaty, kamery czy telefony komórkowe. Bardzo dobrym przykładem jej zastosowania jest projekt DARPA Grand Challenge, w którym wykorzystano bibliotekę OpenCV do przetwarzania i analizy obrazów z kamer autonomicznie poruszającego się samochodu. [1]

Algorytmy śledzenia obiektów zawarte są w pod-bibliotece cxcore i głównie ona będzie przeze mnie wykorzystywana podczas testów. Operacje zawarte w innych bibliotekach są przydatne przy wszelkich modyfikacjach.

Biblioteka OpenCV zawiera kilka funkcji za pomocą, których można śledzić obiekty. Można do nich zaliczyć:

- meanshift
- camshift
- haartraining
- blobs

4.2.1 Camshift

Algorytm Camshift działa w oparciu o metodę ciągłej adaptacji przesunięcia jądra opisanej w rozdziale 3.5.3. Jest on jedną z podstawowych funkcji śledzących obiekty w bibliotece OpenCV. Jej postać jest rozszerzeniem funkcji Mean-Shift o parametr zawierający wymiary okna: [1]

Funkcja Camshift w bibliotece OpenCV zapisana jest w postaci: [1]

```
int cvCamShift
(
constCvArr* prob_image,
CvRect window,
CvTermCriteria criteria,
CvConnectedComp*comp,
CvBox2D* box = NULL
);
```

gdzie:

prob_image – funkcja prawdopodobieństwa występowania poszczególnych pikseli;
window – pozycja oraz wymiary początkowego okna;
criteria – kryterium zatrzymania algorytmu;
comp – zawiera wynikowe okno skoordynowane z przesunięciem obiektu oraz sumę pikseli w nim zawartych;
box – zawiera parametry okna, obiektu z poprzednich obliczeń;

Jako funkcje prawdopodobieństwa przyjęty został obraz rozkładu prawdopodobieństwa występowania określonych wartości pikseli obrazu na podstawie histogramu (ang. backproject) [1]. Na podstawie obrazu wzorcowego oraz wybranego rozkładu kolorów tworzony jest jednokanałowy obraz 8b o wymiarach zgodnych z rozmiarem badanej sekwencji. Znajdujące się na nim obszary jaśniejsze określają wyższą wartość funkcji prawdopodobieństwa natomiast ciemniejsze niższą.

Ilustruje to poniższy Rysunek 46.przedstawiający przykładowy model, określony rozkład kolorów oraz obraz „backproject”.

a) b)

Rysunek 46. Przykład a) badanego obiektu i odpowiadającemu jemu b) obrazu „backproject”

Działanie algorytmu Camshift zostało sprawdzone na sekwencjach przedstawionych w rozdziale 4.1. Jako parametry wejściowe użyte zostały:

- obraz „backproject” uzyskany na podstawie dwukanałowego obrazu zawierającego składowe H i S obrazu wzorcowego oraz histogramu 2 wymiarowego badanego obiektu (uzyskanego również ze składowych H i S) ;
- wymiary okna badanego obiektu, aktualizowane dla kolejnych obrazów sekwencji;
- kryteria zatrzymania algorytmu:

cvTermCriteria(CV_TERMCRIT_EPS | CV_TERMCRIT_ITER, 10, 1)

gdzie:

CV_TERMCRIT_ITER – kryterium maksymalnej liczby iteracji = 10;

CV_TERMCRIT_EPS – kryterium wartości epsilon, limit błędu =1;

- na podstawie wynikowego okna algorytmu Camshift aktualizowana jest pozycja badanego obiektu;

Poniżej przedstawione zostało działanie algorytmu Camshift dla kilku wybranych sekwencji wykorzystywanych podczas badań:

a) b) c)

d) e) f)

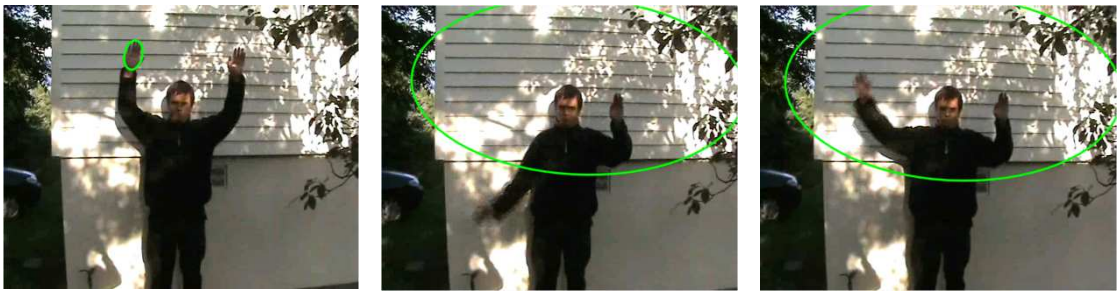
Rysunek 47. Wyniki śledzenia przy użyciu algorytmu Camshift dla Sekwencji 1

a) b) c)

d) e) g)

Rysunek 48. Wyniki śledzenia przy użyciu algorytmu Camshift dla Sekwencji 3

a) b) c)



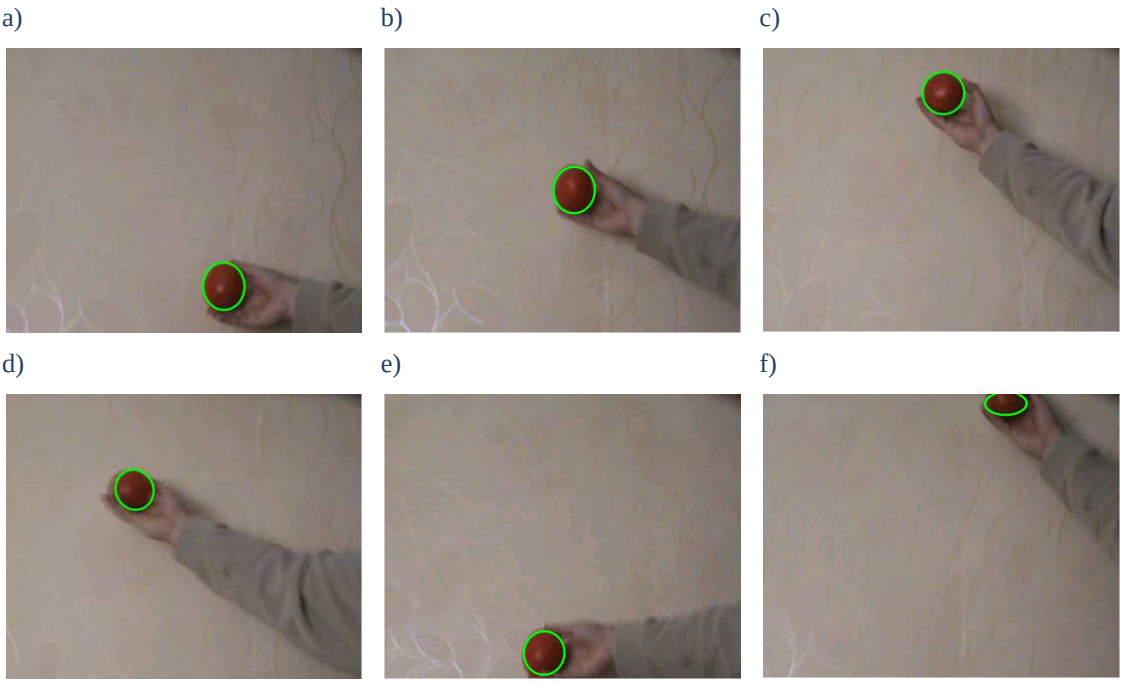
d) e) f)

Rysunek 49. Wyniki śledzenia przy użyciu algorytmu Camshift dla Sekwencji 6

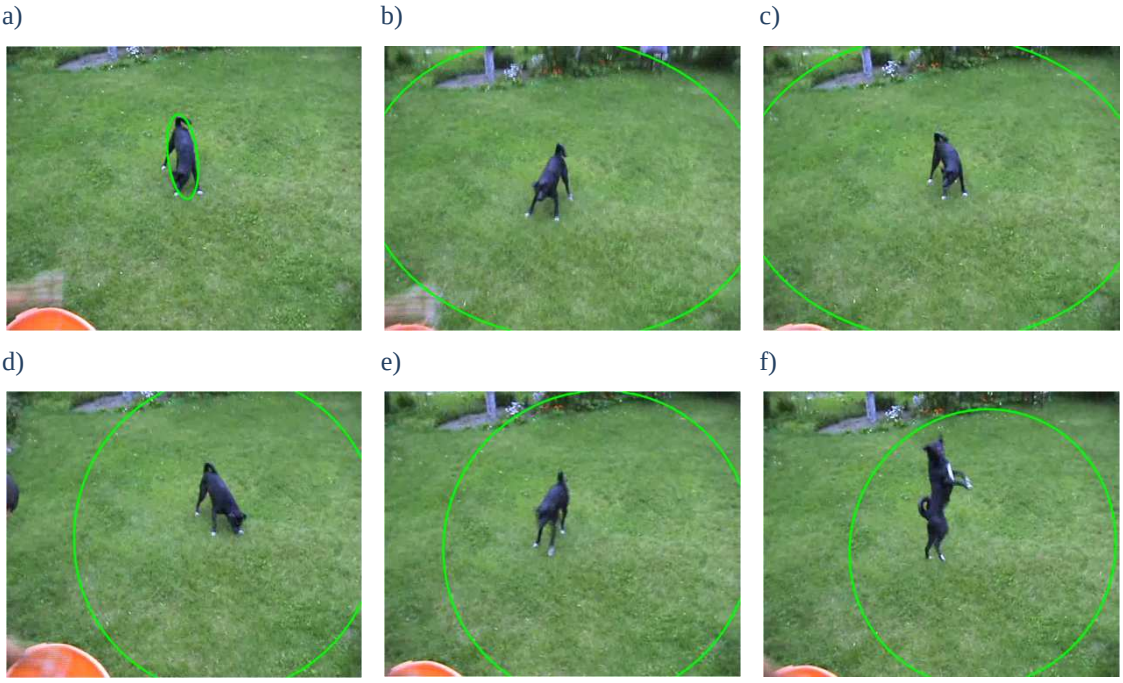
a) b) c)

d) e) f)

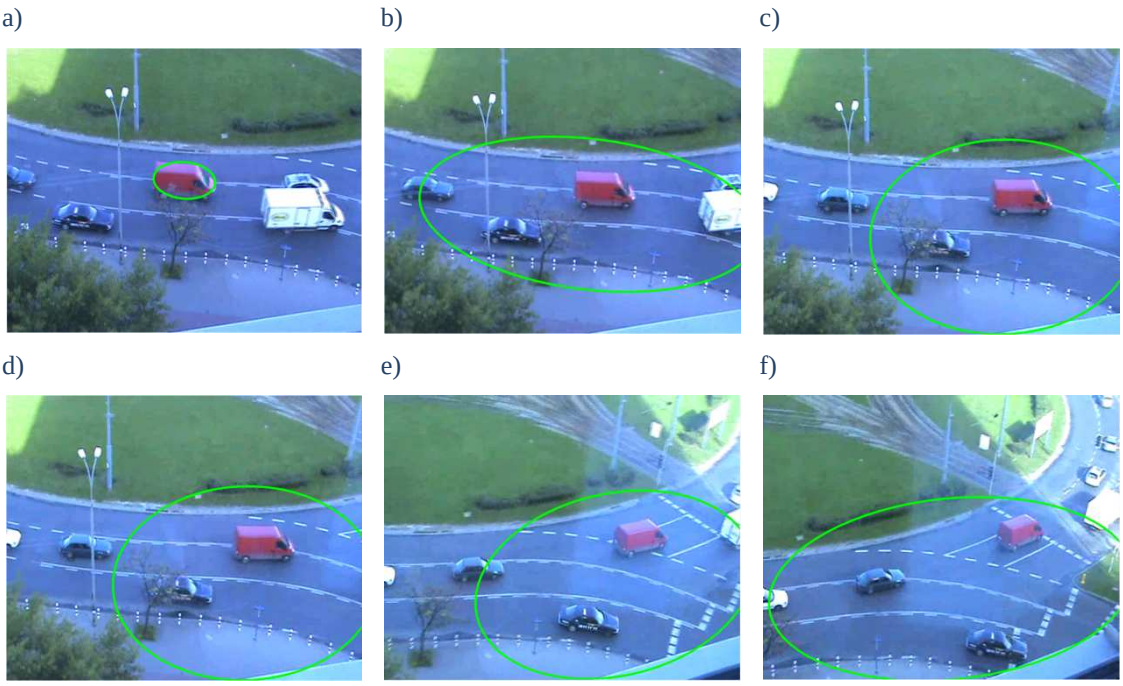
Rysunek 50. Wyniki śledzenia przy użyciu algorytmu Camshift dla Sekwencji 8



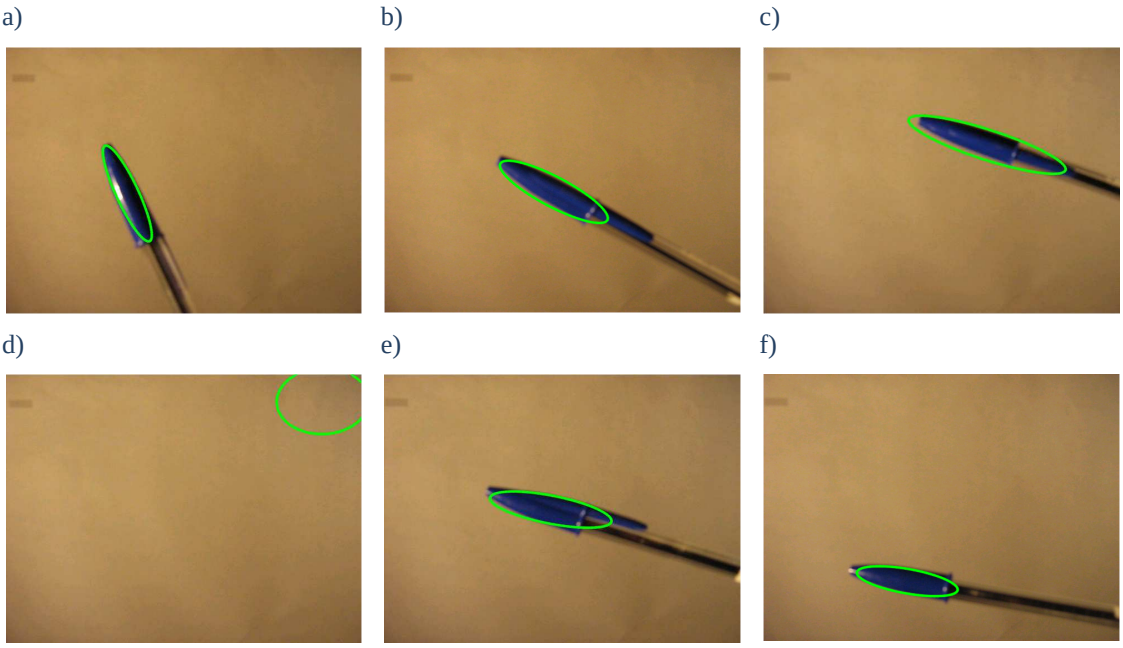
Rysunek 51. Wyniki śledzenia przy użyciu algorytmu Camshift dla Sekwencji 9



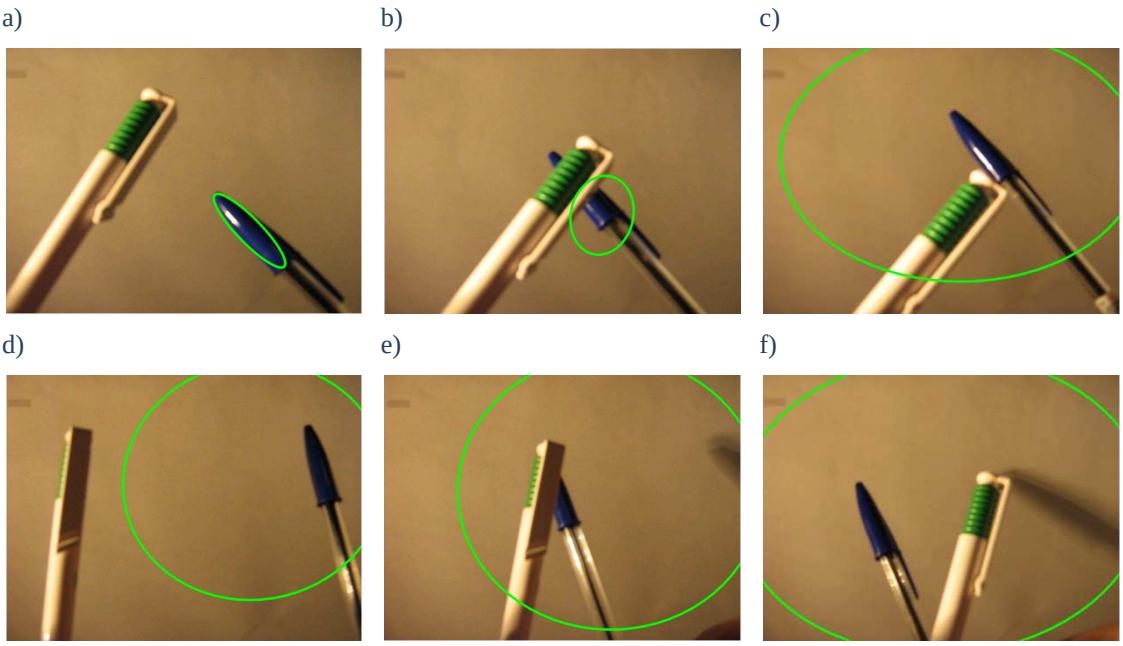
Rysunek 52.Wyniki śledzenia przy użyciu algorytmu Camshift dla Sekwencji 11



Rysunek 53. Wyniki śledzenia przy użyciu algorytmu Camshift dla Sekwencji 12



Rysunek 54. Wyniki śledzenia przy użyciu algorytmu Camshift dla Sekwencji 14



Rysunek 55. Wyniki śledzenia przy użyciu algorytmu Camshift dla Sekwencji 15

Wyniki badań śledzenia obiektów dla algorytmu Camshift zawiera poniższa tabela:

Nr. sekwencji	Średni czas przetwarzania [ms]	Skuteczność działania [%]
1	5	88
2	3	1
3	6	18
4	5	100
5	6	1
6	7	1
7	5	7
8	5	3
9	4	100
10	7	1
11	7	1
12	6	1
13	5	2
14	5	55
15	5	6

Tabela 1. Wyniki badań śledzenia obiektów dla algorytmu Camshift

Na podstawie powyższych badań zaobserwowany został stosunkowo niewielki wpływ rodzaju obiektu oraz sekwencji na średni czas rozpoznania obiektu. Dla wszystkich badanych sekwencji oscylował on w granicach 4 - 7 ms, co jest bardzo dobrym wynikiem i daje możliwość śledzenie obiektów w czasie rzeczywistym. Mimo tego, że algorytm działa szybko, to sprawdza się on jedynie w odpowiednio dobranych sekwencjach. Gdy obiekt jest odmienny od tła, czyli kolorystyka obiektu jest w znacznym stopniu unikatowa algorytm działa dobrze tak jak dla sekwencji 1 (Rysunek 47), 4 oraz 9 (Rysunek 51). Wtedy również nie mają wpływu na jego działanie takie zjawiska jak obrót obiektu, jego oddalenie czy przybliżenie. Odmienna sytuacja została zaobserwowana dla obiektu posiadającego częściowo kolorystykę tła, w którego pobliżu znajdują się obiekty zbliżone do wzorcowego (sekwencja 3, Rysunek 48) jak również dla obiektu o nieregularnym kształcie uniemożliwiającym prawidłowe jego oznaczenie (sekwencja 11, Rysunek 52). W sekwencji 3 algorytm zamiast lokalizować jedynie obiekt, lokalizował również fragment tła, co dało skuteczność działania na poziomie 18%. Dla sekwencji 11 od razu okno wynikowe algorytmu zaczęło się rozszerzać aż do rozmiarów sekwencji, co jest całkowicie błędnym rezultatem. Identycznie zachował się również gdy w sekwencji wystąpiły znaczne zaszumienia (np. sekwencja 6, Rysunek 49) oraz gdy obiekt został przysłonięty (sekwencja 15, Rysunek 55). Camshift wykazał większą skuteczność, gdy obiekt wyszedł poza badane obrazy, lecz jej wartość na poziomie 55% nie jest wystarczająco duża, aby uznać to za dobry wynik (sekwencja 14, Rysunek 54). Algorytm w tym wypadku dobrze zadziałał przy obrocie obiektu natomiast, gdy znalazł się on poza obrazem nadal go lokalizował, co jest nieprawidłowe.

4.2.2 Template Matching

Algorytm ten jest oparty o wcześniej opisaną w rozdziale 3.5.1 metodę dopasowania do szablonu. Fragmenty obrazu, na którym lokalizowany jest szukany obiekt porównywany jest z wzorcem wedle określonej metody (rodzaje metod opisane poniżej).

Funkcja Match Template w bibliotece OpenCV zapisana jest w postaci: [1]

```
void cvMatchTemplate
(
constCvArr* image,
constCvArr* templ,
CvArr* result,
intmethod
);
```

gdzie:
image – obraz wejściowy;
templ – rozpoznawany obiekt;
result – obraz wynikowy;
method – metoda porównania;

Kolejne piksele z obrazu *temp1* porównywane są z pikselami modelu *image* zgodnie z wybraną metodą. Wybierany jest ten fragment, w którym zaklasyfikowanych zostało najwięcej pikseli. Dzięki temu uzyskiwany jest obraz wynikowy *result* zawierający rozpoznaną część obrazu. Na jej podstawie przy użyciu funkcji *cvMinMaxLoc* można odczytać lokalizację wybranego obiektu.

Klasyfikacja poszczególnych pikseli, jako właściwie rozpoznanych może odbywać się na kilka sposobów. Biblioteka OpenCV udostępnia kilka metod porównujących:

CV_TM_SQDIFF - metoda różnicy kwadratów;
CV_TM_CCORR - metoda korelacji;
CV_TM_CCOEFF - metoda korelacji ko efektywnej;
CV_TM_SQDIFF_NORMED, CV_TM_CCORR_NORMED;
CV_TM_CCOEFF_NORMED - metody znormalizowane;

Powyższe metody zostały dokładniej opisane w rozdziale 3.5.1.

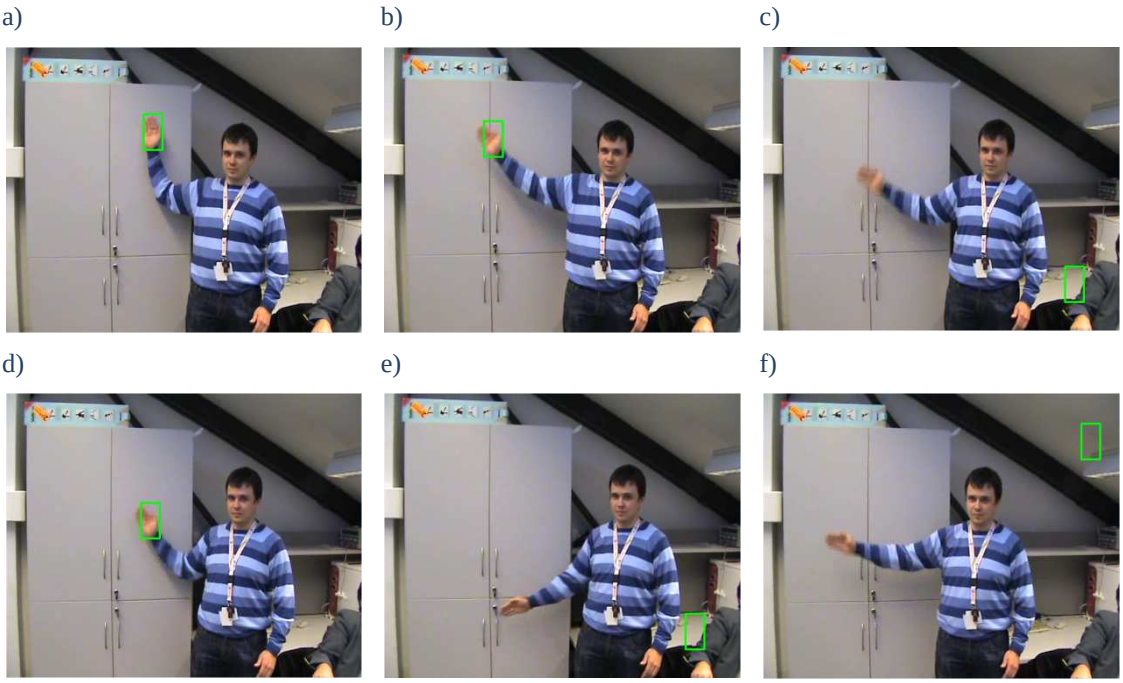
Algorytm Template Matching zaimplementowany w bibliotece OpenCV w celu lokalizacji obiektu przeszukuje cały obraz, co w znacznym stopniu spowalnia jego działanie, gdyż wykonywane są dodatkowe porównania. Aby podczas badań pokazać bardziej zaawansowaną postać tej metody wykorzystany został algorytm „Fast Template Matching" opisany w artykule [35]. Jego postać przedstawia poniższa definicja funkcji:

```
void FastMatchTemplate
(
constCvArr*source,
constCvArr* target,
vector<CvPoint> *foundPointsList,
vector<double>* confidencesList,
intmatchPercentage,
boolfindMultipleTargets,
intnumMaxima,
intnumDownPyrs,
intsearchExpansion)
);
```

gdzie:
source - obraz wejściowy;
target - rozpoznawany obiekt;
foundPointsList – lista wynikowych lokalizacji;
confidencesList–lista wartości zgodności fragmentu z wzorcem;
matchPercentage - procent zgodności fragmentu z wzorcem =70;
matchMultipleTargets- możliwość śledzenia wielu obiektów na raz = false;
numMaxima - ilość możliwych maksimów dla algorytmu =5;
numDownPyrs - wartość skalująca obraz do przeszukiwań = 2;
searchExpansion– obszar pikseli przeszukiwany wokół otrzymanych maksimów= +/- 15;

Funkcja Fast Match Template w pierwszej kolejności wyznacza listę punktów lokalizacji obiektów *foundPointsList* oraz odpowiadające im wartości zgodności *confidencesList*. Do ich wyboru wykorzystana została funkcja cvMatchTemplate z

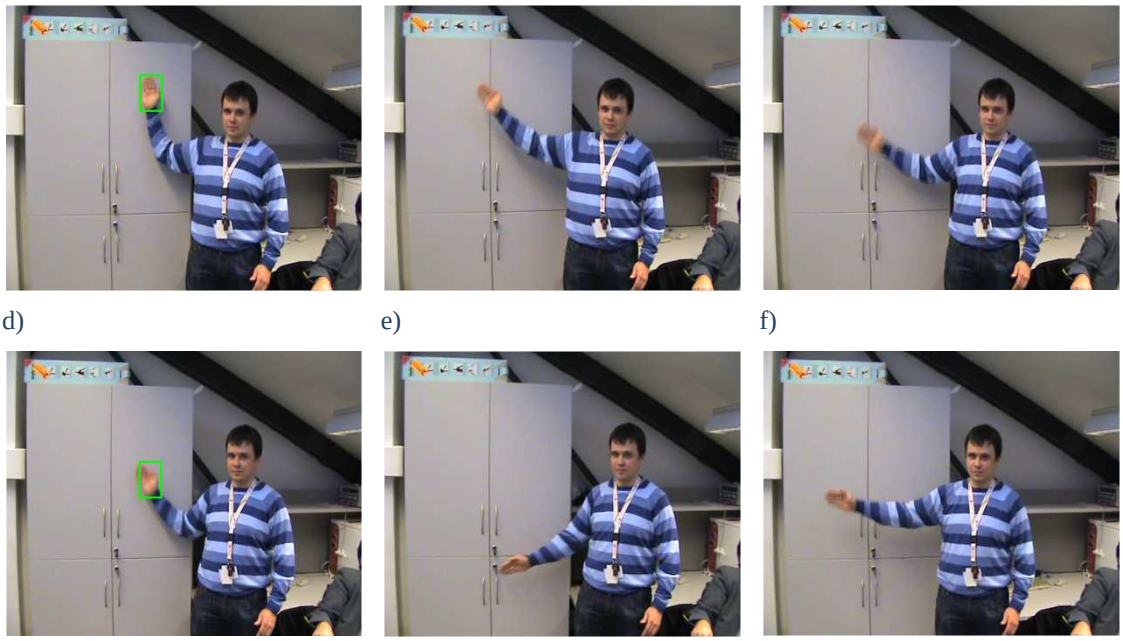
użyciem metody porównawczej *CV_TM_CCOEFF_NORMED*. Otrzymane punkty zapisywane są jedynie wtedy, gdy wartość dopasowania przekracza przyjętą wartość minimalną *matchPercentage* oraz gdy ich ilość nie jest większa niż parametr *numMaxima*. Od tego momentu nie jest już konieczne sprawdzanie całego obrazu tak jak w wypadku zaimplementowanego algorytmu Template Matching, lecz jedynie obszaru ±*searchExpansion* pikseli wokół wybranych punktów *foundPointsList*. Dzięki takiemu podejściu algorytm „Fast Template Matching” jestblisko 5 razy szybszy od podstawowej wersji zaimplementowanej w bibliotece OpenCV, co zostało sprawdzone dla sekwencji 1 z Rysunku 31. Średni czas rozpoznania dla funkcji cvMatchTemplate wyniósł w okolicach 190ms natomiast dla Fast Template Matching ok. 40 ms. Istotne jest również to, że podstawowa funkcja nie odrzuca obiektów o zbyt niskim dopasowaniu co wprowadza błędy. Ilustruje to poniższy rysunek: [35]



Rysunek 56.Wyniki śledzenia przy użyciu algorytmu „TemplateMatching” dla Sekwencji 1

Poniżej przedstawione zostało działanie algorytmu „Fast Template Matching” dla kilku wybranych sekwencji wykorzystywanych podczas badań:

a) b) c)



d) e) f)

Rysunek 57. Wyniki śledzenia przy użyciu algorytmu „Fast Template Matching” dla Sekwencji 1

a) b) c)

d) e) f)

Rysunek 58.Wyniki śledzenia przy użyciu algorytmu „Fast Template Matching” dla Sekwencji 3

a) b) c)

d) e) f)

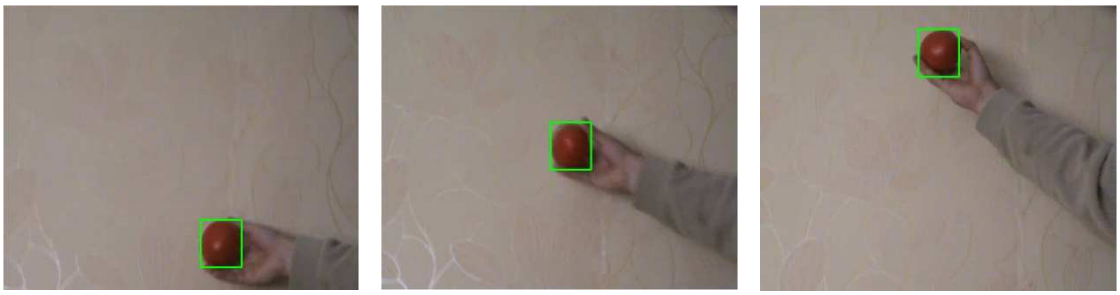
Rysunek 59.Wyniki śledzenia przy użyciu algorytmu „Fast Template Matching” dla Sekwencji 6

a) b) c)

d) e) f)

Rysunek 60.Wyniki śledzenia przy użyciu algorytmu „Fast Template Matching” dla Sekwencji 8

a) b) c)



d) e) f)

Rysunek 61. Wyniki śledzenia przy użyciu algorytmu „Fast Template Matching” dla Sekwencji 9

a) b) c)

d) e) f)

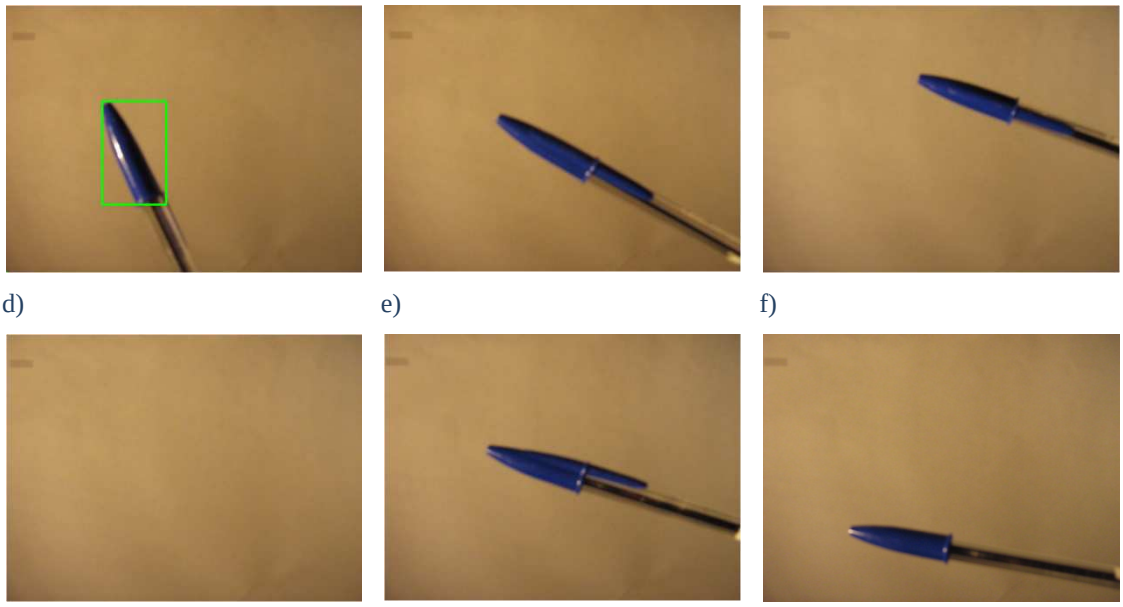
Rysunek 62. Wyniki śledzenia przy użyciu algorytmu „Fast Template Matching” dla Sekwencji 11

a) b) c)

d) e) f)

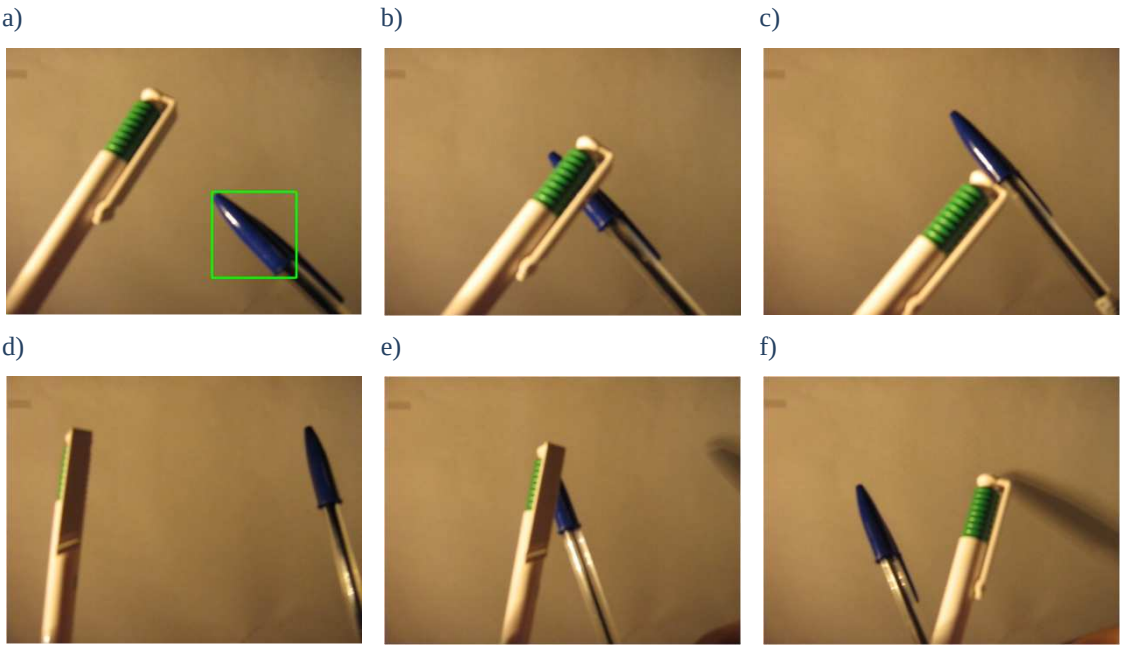
Rysunek 63. Wyniki śledzenia przy użyciu algorytmu „Fast Template Matching” dla Sekwencji 12

a) b) c)



d) e) f)

Rysunek 64. Wyniki śledzenia przy użyciu algorytmu „Fast Template Matching” dla Sekwencji 14



Rysunek 65. Wyniki śledzenia przy użyciu algorytmu „Fast Template Matching” dla Sekwencji 15

Wyniki badań śledzenia obiektów dla algorytmu „Fast Template Matching”:

Nr. sekwencji	Średni czas przetwarzania [ms]	Skuteczność działania [%]
1	45	24
2	19	100
3	51	86
4	28	100
5	39	3
6	40	2
7	47	3
8	32	72
9	39	88
10	43	80
11	66	4
12	49	51
13	45	17
14	71	27
15	70	22

Tabela 2. Wyniki badań śledzenia obiektów dla algorytmu „Fast Template Matching”

Na podstawie powyższych badań zaobserwowane zostały stosunkowo duże różnice pod względem średniego czasu rozpoznania w zależności od badanej sekwencji. Najwyższy czas zmierzony został dla sekwencji 14 (Rysunek 74) i wyniósł 71 ms, natomiast najkrótszy dla sekwencji 2 wyniósł 19 ms, co ilustruje również Tabela 2. Takie rozbieżności są spowodowane takimi czynnikami jak: wielkość obiektu, prędkość zmiany położenia oraz różnego rodzaju zaszumienia. Czas rozpoznania nie wpływa ani pozytywnie ani negatywnie na skuteczność działania algorytmu. Ma jednak duży wpływ zaszumienie obrazów, co zostało zaobserwowane np. dla sekwencji 6 (Rysunek 59) dla której prawidłowość lokalizacji wyniosła jedynie 2%. Jest to spowodowane dużymi zmianami kolorystyki obiektu pod względem szybkiego ruchu, zmiennego oświetlenia i dużego zaszumienia sekwencji. Gdy na obiekt nie wpływają inne czynniki zewnętrzne algorytm pod względem prawidłowości rozpoznania działa bardzo dobrze. Przykładem tego są sekwencje 4 oraz 9 (Rysunek 61), dla których skuteczność rozpoznania wyniosła odpowiednio 100% i 88%.Trzeba oczywiście pamiętać, że dla tych sekwencji obiekt nie ulegał obrotowi, przybliżeniu ani oddaleniu. Działanie algorytmu dla takiego ruchu obiektu przedstawiają Rysunek 57 czy Rysunek 64. Skuteczność działania metody „Fast Template Matching” dla obu tych przypadków była niska - poniżej 30%. Podobne rezultaty zostały uzyskane, gdy obiekt podczas trwania sekwencji wyszedł poza jej obraz czy również został przysłonięty przez inny obiekt.

4.2.3 *Przepływ optyczny*

Algorytm przepływu optycznego działa w oparciu o metodę opisaną w rozdziale 3.2.2. Badany jest w niej ruch pikseli podczas trwania sekwencji. W bibliotece OpenCV zaimplementowanych zostało kilka algorytmów przepływu optycznego:

- CalcOpticalFlowHS – wykorzystujący algorytm Horn & Schunck, w którym są wykonywane obliczenia z wykorzystaniem pochodnych wyższego rzędu.
- CalcOpticalFlowBM – wykorzystujący porównywanie blokowe zbiorów pikseli np. 3x3 (ang. Block Matching);

- CalcOpticalFlowLK – wykorzystujący algorytm Lucas-Kanade, w której porównywane są poszczególne piksele obrazu;
- CalcOpticalFlowPyrLK – wykorzystujący algorytm Lukas-Kanade z zastosowaniem piramid Gaussa, umożliwiający porównywanie wybranych pikseli obrazów oraz wykrywanie większych niż w metodzie CalcOpticalFlowLK ruchów obiektu;

Ze względu na największy stopień zaawansowania, ilość parametrów funkcji, możliwość wyboru punktów do śledzenia oraz przydatności do badań została wykorzystana metoda CalcOpticalFlowPyrLK. Funkcja CalcOpticalFlowPyrLK w bibliotece OpenCV zapisana jest w postaci: [1]

```
void cvCalcOpticalFlowPyrLK
(
constCvArr* prev,
constCvArr* curr,
CvArr* prev_pyr,
CvArr* curr_pyr,
const CvPoint2D32f* prev_features,
CvPoint2D32f* curr_features,
int count,
CvSizewin_size,
int level,
char* status,
float* track_error,
CvTermCriteria criteria,
int flags
);
```

gdzie:

prev – pierwsza klatka sekwencji w czasie t;

curr – druga klatka sekwencji w czasie t + dt;

prev_pyr – buffor dla piramidy z pierwszej klatki sekwencji;

curr_pyr – buffor dla piramidy z drugiej klatki sekwencji;

prev_features– wektor zawierający punkty, dla których wyznaczany będzie przepływ optyczny;

curr_features – wektor 2D zawierający kalkulacje nowych pozycji punktów;

count – maksymalna liczba punktów do sprawdzenia = 2400;

win_size – wielkość okna przeszukiwań dla każdego poziomu piramidy = 3x3;

level – maksymalny poziom piramidy = 5;

status – wektor zawierający statusy 0/1 określające znalezienie danego punktu;

error –wektor zawierający różnicę pomiędzy oryginalnymi a przemieszczonymi punktami;

criteria - kryterium zatrzymania algorytmu = cvTermCriteria(CV_TERMCRIT_ITER | CV_TERMCRIT_EPS, 20, .3);

flags – flaga różnic = 0;

CV_LKFLOW_PYR_A_READY – piramida dla klatki 1 sekwencji jest wyznaczania wcześniej;

CV_LKFLOW_PYR_B_READY - piramida dla klatki 2 sekwencji jest wyznaczania wcześniej;

CV_LKFLOW_INITIAL_GUESSES – macierz B zawiera wstępne położenie punktów;

Algorytm ten na podstawie dwóch kolejnymi klatek sekwencji *prev* i *curr* wyznacza nowe pozycje wybranych punktów obrazów *prev_features*, których współrzędne umieszczane są w zmiennej *curr_features*.Rozpoznanie jest wykonywane zarówno na badanych obrazach jak iich przekształceniach zgodnym z piramidą Gaussa (opisaną w rozdziale 3.4.3). Jeśli któryś z punktów nie zostanie zlokalizowany jest on oznaczany flagą *status*. Aby takich sytuacji było jak najmniej badane punkty muszą być jak najbardziej unikatowe i dlatego do ich wyznaczenia wykorzystana została funkcja cvGoodFeaturesToTrack, która w bibliotece OpenCV zapisana jest w postaci: [1]

```
void cvGoodFeaturesToTrack(
constCvArr* image,
CvArr* eig_image,
CvArr* temp_image,
CvPoint2D32f* corners,
int* corner_count,
doublequality_level,
doublemin_distance,
constCvArr* mask=NULL,
intblock_size=3,
intuse_harris=0,
double k=0.04,
);
```

gdzie:
image – jedno - kanałowy obraz wejściowy;
eig_image –obraz tymczasowy o wymiarach obrazu image;
temp_image – dodatkowy obraz tymczasowy zgodny z obrazem eig_image;
corners – parametr wyjściowy zawierający lokalizacje narożników;
corner_count– parametr wyjściowy zwracający liczbę wybranych narożników;
quality_level– minimalna wartość narożnika = 0.1;
min_distance– minimalny dystans pomiędzy wybieranymi narożnikami = 0.1;
mask– maska ograniczająca obszar obrazu;
block_size– minimalny rozmiar bloku uśredniającego = 3x3;
use_harris– używany operator 0 – cvCornerMinEigenVal,inny – cvCornerHarris = 0;
k– wolny parametr dla detektora Harrisa;

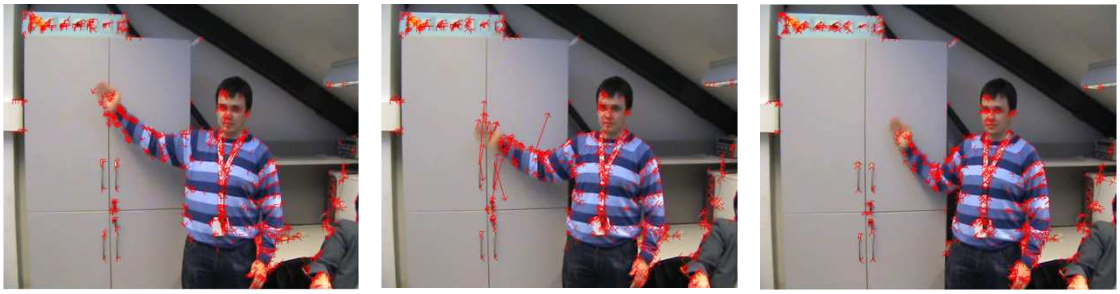
Funkcja cvGoodFeaturesToTrackwykorzystyje opisane w rozdziale 3.4.2 „cechy dobre do śledzenia”. Znajduje ona pozycje narożników z największym znaczeniem dla całego obrazu. W pierwszej kolejności wyznaczany jest wartość minimalnego narożnika *quality_level* za pomocą funkcji cvCornerMinEigenVal, po czym na jej podstawie odrzucane są wszystkie punkty, których wartość jest mniejsza od wartości minimalnej. Oprócz tego eliminowane są również narożniki, które są od siebie oddalone o mnie niż *min_distance* pikseli, przy czym w pierwszej kolejności sprawdzane są te o wyższej wartości. Tak otrzymana lista punktów „dobrych do

śledzenia" wykorzystywana jest, jako parametr wejściowy dla funkcji cvCalcOpticalFlowPyrLK.

Podczas badań w pierwszej kolejności na podstawie poprzedniej klatki wyznaczana jest lista punktów „dobrych do śledzenia" (cvGoodFeaturesToTrack). W kolejnym kroku przy użyciu cvCalcOpticalFlowPyrLK określane jest ich położenie w kolejnej klatce sekwencji. Na podstawie uzyskanych wyników liczone jest średnie przemieszczenie pikseli dla wszystkich punktów znajdujących się w obszarze badanego obiektu (przy czym ze względu na wprowadzane zakłócenia eliminowane są ruchy poniżej 10 pikseli). Otrzymany wektor przesunięcia dodawany jest do poprzedniej pozycji okna obiektu, dzięki czemu uzyskiwane jest bieżące położenie obiektu. Ze względu na zmiany wartości pikseli pod wpływem ruchu obiektu, czy oświetlenia w kolejnych klatkach sekwencji, punkty do badań są wyznaczane dla każdego kolejnego porównania.

Poniżej umieszczony został przykład działania funkcji cvCalcOpticalFlowPyrLK z wykorzystaniem cvGoodFeaturesToTrack do uzyskania punktów do badań ruchu. Na czerwono oznaczone zostały wybrane punkty oraz kierunek ich przemieszczenia.

a) b) c)



Rysunek 66. Przykład działania funkcji cvCalcOpticalFlowPyrLK z wykorzystaniem „cech dobrych do śledzenia"

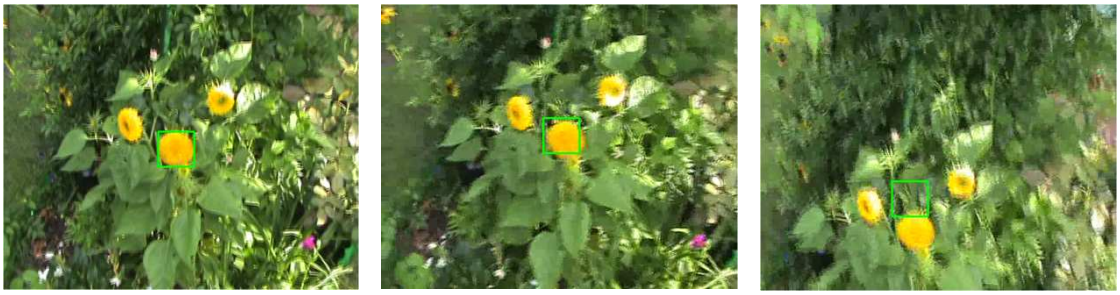
Poniżej przedstawione zostało działanie algorytmu „Pyramid Lukas-Kanade Optical Flow" dla kilku wybranych sekwencji wykorzystywanych podczas badań:

a) b) c)

d) e) f)

Rysunek 67. Wyniki śledzenia przy użyciu algorytmu „Pyramid Lukas-Kanade Optical Flow” dla Sekwencji 1

a) b) c)



d) e) f)

Rysunek 68.Wyniki śledzenia przy użyciu algorytmu „Pyramid Lukas-Kanade Optical Flow” dla Sekwencji 3

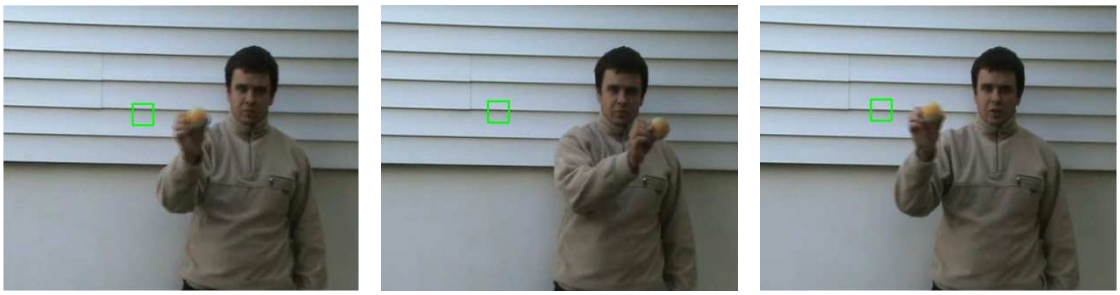
a) b) c)

d) e) f)

Rysunek 69.Wyniki śledzenia przy użyciu algorytmu „Pyramid Lukas-Kanade Optical Flow” dla Sekwencji 6

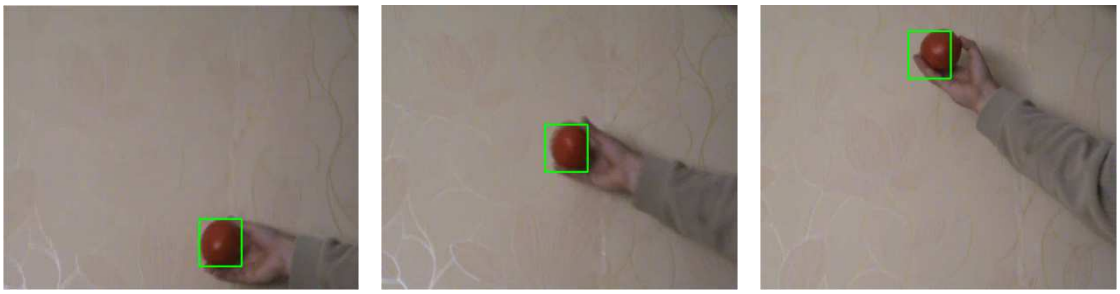
a) b) c)

d) e) f)

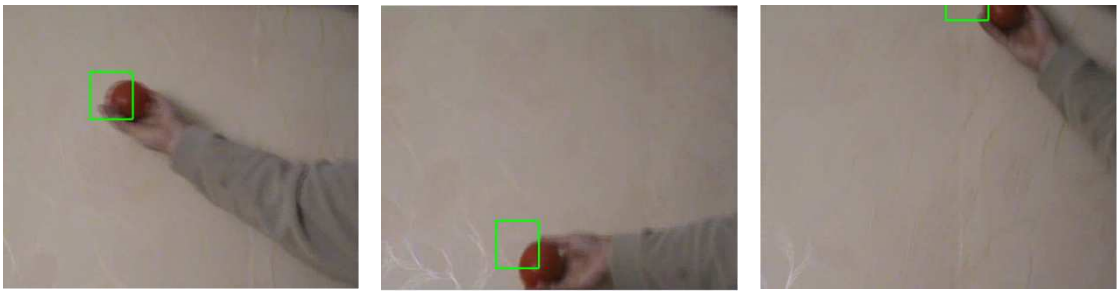


Rysunek 70.Wyniki śledzenia przy użyciu algorytmu „Pyramid Lukas-Kanade Optical Flow” dla Sekwencji 8

a) b) c)



d) e) f)



Rysunek 71. Wyniki śledzenia przy użyciu algorytmu „Pyramid Lukas-Kanade Optical Flow” dla Sekwencji 9

a) b) c)

d) e) f)

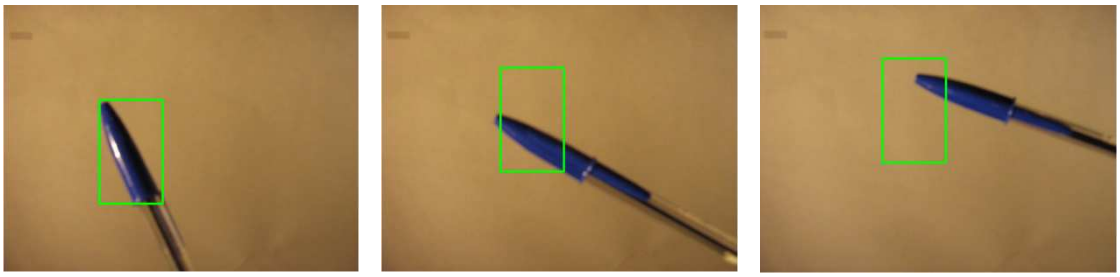
Rysunek 72. Wyniki śledzenia przy użyciu algorytmu „Pyramid Lukas-Kanade Optical Flow” dla Sekwencji 11

a) b) c)

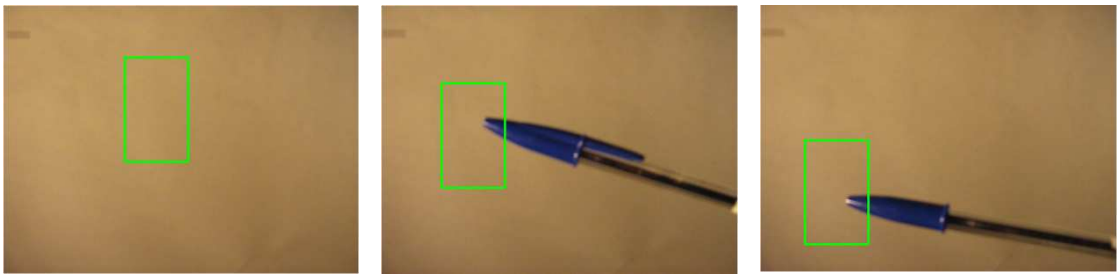
d) e) f)

Rysunek 73. Wyniki śledzenia przy użyciu algorytmu „Pyramid Lukas-Kanade Optical Flow” dla Sekwencji 12

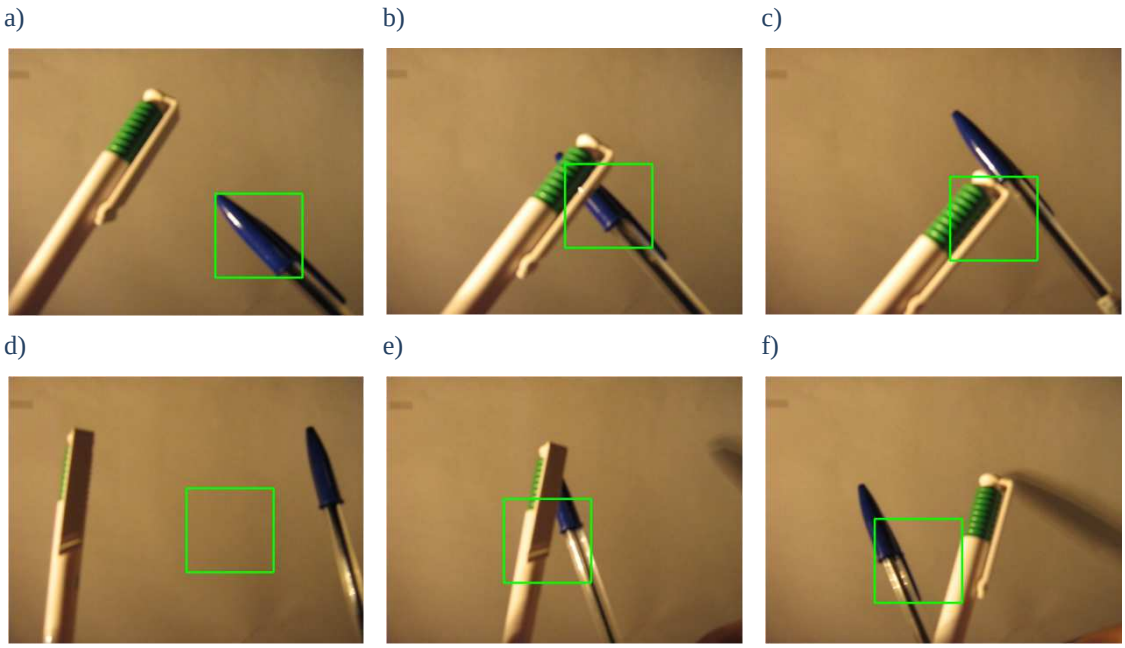
a) b) c)



d) e) f)



Rysunek 74. Wyniki śledzenia przy użyciu algorytmu „Pyramid Lukas-Kanade Optical Flow” dla Sekwencji 14



Rysunek 75. Wyniki śledzenia przy użyciu algorytmu „Pyramid Lukas-Kanade Optical Flow" dla Sekwencji 15

Wyniki badań śledzenia obiektów dla algorytmu przepływu optycznego:

Nr. sekwencji	Średni czas przetwarzania [ms]	Skuteczność działania [%]
1	60	19
2	34	61
3	73	24
4	68	12
5	60	1
6	61	3
7	61	2
8	64	10
9	65	35
10	68	7
11	64	34
12	77	12
13	62	7
14	60	16
15	58	16

Tabela 3.Wyniki badań śledzenia obiektów dla przepływu optycznego metodą „Pyramid Lukas-Kanade Optical Flow" (cvCalcOpticalFlowPyrLK) z wykorzystaniem dobrych do śledzenia cech (cvGoodFeaturesToTrack)

Na podstawie powyższych badań zauważyć można zbliżony średni czas rozpoznania oraz niską skuteczność działania algorytmu dla większości sekwencji niezależnie od ich typu. Algorytm nie nadążał za ruchem obiektu, co widać dla sekwencji 1 (Rysunek 67), 3 (Rysunek 68) czy 9 (Rysunek 71). Jeszcze gorsze rezultaty uzyskane zostały przy dużym zaszumieniu obrazów (sekwencje 6 (Rysunek 69), 7 czy 8 (Rysunek 70)), czy również zmianie orientacji (sekwencja 14 (Rysunek 74)) oraz przysłonięciu (sekwencja 15 (Rysunek 75)). Dla tego rodzaju sekwencji algorytm pozostał w jednym punkcie mimo ruchu obiektu. Najlepsze rezultaty uzyskał dla sekwencji 2, lecz było to spowodowane niewielkim przemieszczeniem obiektu, przez co prawidłowość rozpoznania na poziomie 61% nie jest satysfakcjonująca. Najniższa skuteczność 1% została zbadana dla sekwencji, 5 co jest wynikiem dużych zaszumień obrazu.

4.3 Podsumowanie badań

Na podstawie powyższych badań trzech wybranych metod śledzenia obiektów zauważyć można, że żadna z nich nie jest optymalna dla wszystkich badanych sekwencji. Najgorsza okazała się metoda przepływu optycznego, która wykorzystuje jedynie niewielką część opisu obiektu,. Prawidłowe rozpoznanie przemieszczenia pojedynczych pikseli, niewielkich obszarów czy wybranie właściwego wektora dla obszaru obiektu jest bardzo problematyczne, gdyż mają na to decydujący wpływ wszelkiego rodzaju zakłócenia zmieniające wartości pikseli. Identyczne działanie zaobserwowane zostało dla algorytmu Camshift. Wszelkiego rodzaju zaszumienia takie jak np. złe oświetlenie, przysłonięcia obiektu powodowały całkowite rozszerzenie się okna obrysowującego obiekt. Jest to spowodowane tym, że algorytm Camshift korzysta w głównej mierze z histogramu obiektu, więc jeśli pojawią się w nim jakiekolwiek zakłócenia lub jest on zbliżony do tła algorytm błędnie rozpoznaje obiekt. Trochę lepiej w tym wypadku zadziałał algorytm „Fast Template Maching”, lecz również jego skuteczność była niska. Gdy obiekt był charakterystyczny w stosunku do tła oraz podczas trwania sekwencji nie miały na niego wpływu czynniki zewnętrzne algorytmy Camshift oraz „Fast Template Matching” prezentowały dużo wyższą skuteczność rozpoznania (nawet 100%) niż przepływ optyczny. Było to spowodowane tym, że dwie pierwsze metody wykorzystują znacznie więcej

informacji o obiekcie, przez co są mniej podatne na zmiany w jego kolorystyce czy kształcie. Dla sekwencji, w których badany element zmieniał orientacje czy oddalał/przybliżał się względem urządzenia rejestrującego jedynie algorytm Camshift właściwie lokalizował obiekt, chociaż na wyniki znaczący wpływ miało prawidłowe jego oznaczenie. Pozostałe metody nie umożliwiają śledzenia takich zmian obiektu.

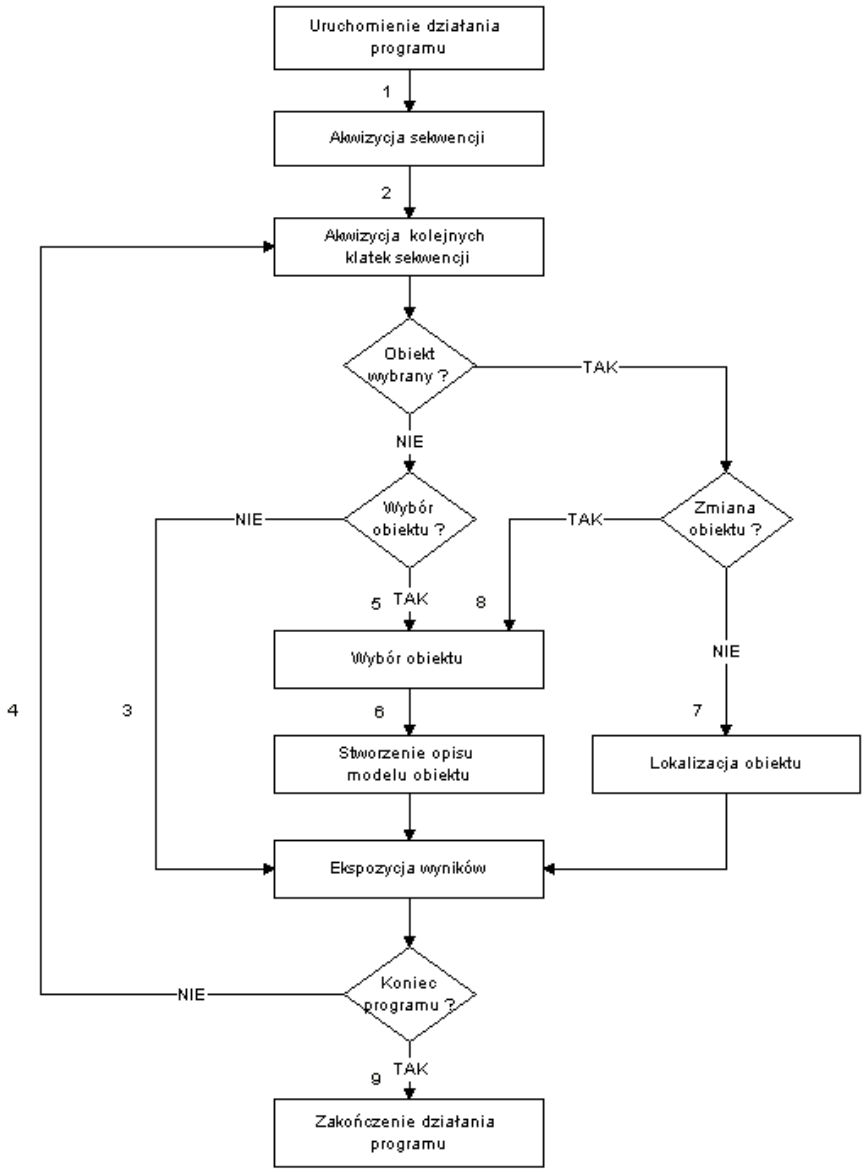
Pod względem czasu rozpoznania najlepszą z badanych funkcji okazał się „Camshift", który jest zdecydowanie najszybszy. Pozostałe algorytmy były ok. 10 razy wolniejsze.

5 Modyfikacja wybranego algorytmu śledzenia obiektów

5.1 Wprowadzenie

Niniejszy rozdział przedstawia proces tworzenia algorytmu śledzenia obiektów w sekwencjach obrazów. Składa się on z wielu kroków, z których można wyróżnić: pozyskanie oraz przygotowanie sekwencji, wyselekcjonowanie wybranego fragmentu obrazu, wyznaczenie „cech" umożliwiających rozpoznanie obiektu. W dalszej części opisany został etap rozwijania algorytmu o kolejne funkcjonalności poprawiające jego działanie.

Pierwszą czynnością jest wczytywanie sekwencji obrazów zawierających badany obiekt. Zostało to zrealizowane poprzez zastosowanie funkcji *cvCaptureFromAVI* korzystającej z filmu w formacie AVI. Schemat działania stworzonego algorytmu przedstawia Rysunek 76. Ze zmiennej, do której wczytano całą sekwencję„1" pobierane są kolejne obrazy „2" (*cvQueryFrame)* a następnie wyświetlane w stworzonym oknie „3". Algorytm pobiera kolejne klatki sekwencji „4" aż do momentu wyboru obiektu „5". Jest to realizowane podczas zatrzymania akwizycji „2". Na podstawie otrzymanych danych tworzony jest model obiektu „6", a następnie na jego podstawie dokonywana jest lokalizacja obiektu w kolejnych klatkach sekwencji „7". W każdym momencie możliwa jest zmiana wyboru obiektu „8" i stworzenia jeszcze raz modelu obiektu. Po każdym z kroków wyświetlane są uzyskane wyniki, po czym albo pobierane są kolejne klatki sekwencji „4" albo następuje koniec programu „9".



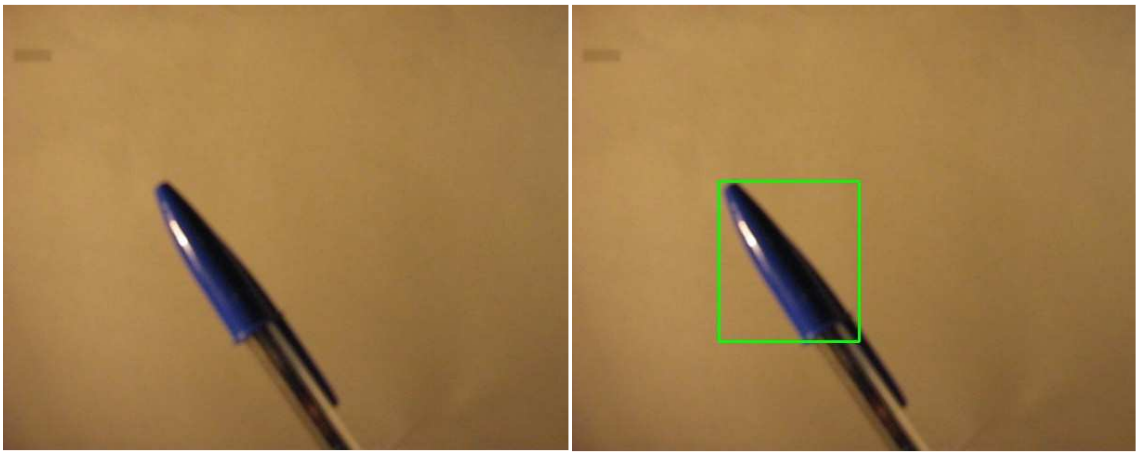
Rysunek 76. Schemat blokowy przedstawiający działanie algorytmu

5.2 Wybór obiektu

Po wczytaniu badanej sekwencji uzyskaniu dostępu do poszczególnych jej klatek kolejnym etapem, istotnym z punktu widzenia pracy jest wybór obiektu, a dokładniej sposób jego zaznaczania. Do tego celu stworzona została funkcja obsługi myszki *on_mouse*. Zwraca ona takie parametry jak położenie kursora czy wymiary zaznaczonego fragmentu, na podstawie, których rysowany jest obrys wybranego obiektu. Ze względu na to, iż zaznaczenie obiektu w ruchu jest dosyć trudne do programu wprowadzone zostały opcje zatrzymania i startu sekwencji obrazów, które w dużym stopniu zwiększają dokładność wyboru obiektu do śledzenia.

5.2.1 Oznaczenie prostokątne

Najprostszym sposobem zaznaczania obiektu jest oznaczanie jego prostokątem. Polega ono na zaznaczeniu punktu początkowego, a następnie przeciągnięciu myszką odpowiedniej wielkości prostokąta tak, aby badany obiekt znajdował się w jego wnętrzu. Taki przykład wyboru obiektu przedstawia Rysunek 77, gdzie badanym obiektem był długopis (dosyć prosty element o małej złożoności i charakterystycznej jednokolorowej skuwce), który został odpowiednio oznaczony poprzez zielony prostokąt.



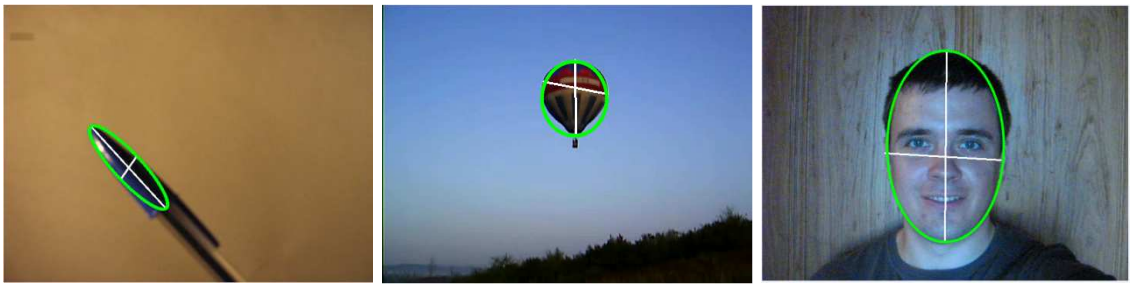
Rysunek 77. Przykład prostokątnego zaznaczania obiektu

Prostokątny sposób oznaczania obiektów jest łatwy zarówno dla użytkownika jak i w implementacji oraz przetwarzaniu obrazów. Przeszukiwanie danej klatki w celu lokalizacji obiektu bardzo często odbywa się poprzez przesuwanie prostokątnego okna wzdłuż obu osi obrazu. Proces porównywania też nie jest zbytnio skomplikowany, gdyż łatwo wyselekcjonować odpowiedniej wielkości fragment odpowiadający rozmiarem do wzorca. Oczywiście taki sposób zaznaczania bez dodatkowych modyfikacji miał jedną podstawową wadę: w obszarze, z którego tworzony był model obiektu oprócz jego samego znajdowała się duża część tła. Miało to duży wpływ na wyniki rozpoznania, a co za tym idzie wprowadzało dodatkowe błędy.

5.2.2 Oznaczenie eliptyczne

Aby udoskonalić działanie algorytmu zmieniony został sposób wyboru obiektu. Do tego celu wykorzystana została elipsa, dzięki której dużo lepiej można oznaczyć wybrany obiekt np. przykładową skuwkę, czy również twarz, owoc, balon, rybki itp. Dzięki temu można uzyskać lepszy opis wzorca, a co za tym idzie zwiększa to prawdopodobieństwo jego właściwego rozpoznania w kolejnych klatkach sekwencji obrazów.

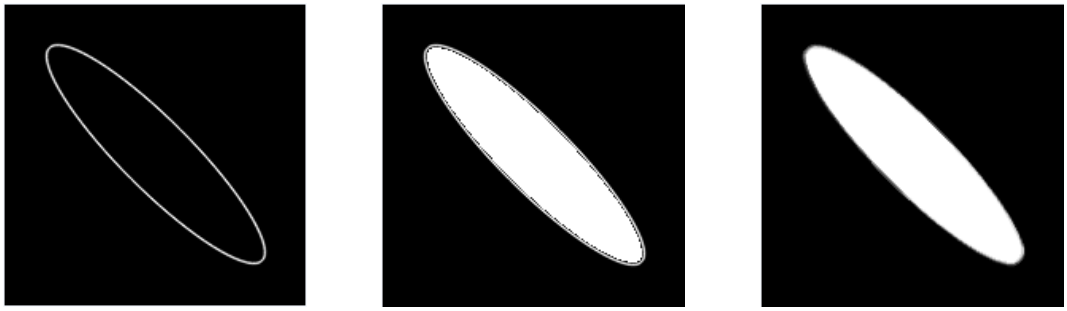
Zaimplementowany sposób zaznaczania obiektu poprzez elipsę polega na: oznaczeniu jednej z przekątnych, z której odczytywana jest jej długość oraz jej kąt położenia, a następnie poprzez narysowanie drugiej średnicy wyznaczana jest długość drugiej przekątnej. Dzięki tym operacjom uzyskiwany jest opis elipsy obrysowującej badany obiekt. Zaznaczenie obiektu tą metodą przedstawia poniższy Rysunek 78.



Rysunek 78. Przykładowe eliptycznego zaznaczania obiektu

Aby wydzielić odpowiedni obszar o kształcie elipsy na podstawie wcześniej uzyskanego opisu tworzona jest maska. Możliwe było przeszukiwanie danej klatki bez tego kroku, ale byłoby to bardzo skomplikowane. Proces jej tworzenia przedstawia Rysunek 79.

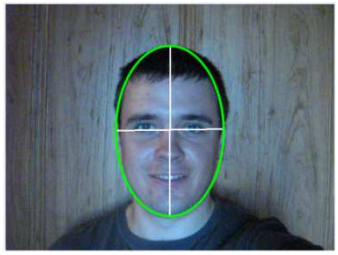
a) b) c)



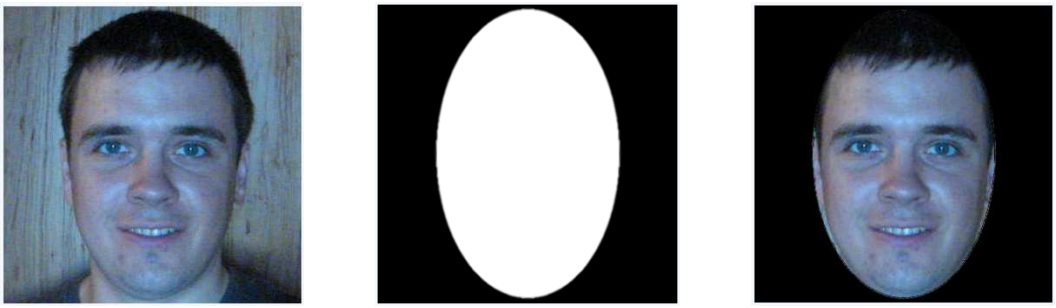
Rysunek 79.Proces tworzenia maski obiektu:
rysowanie elipsy; b) wypełnienie środka elipsy; c) filtracja

W pierwszym etapie na czarnym obrazie rysowana jest biała elipsa o parametrach zgodnych z ustalonymi podczas zaznaczania. Następnie jej wnętrz wypełniane jest kolorem białym wykorzystując funkcję *cvFloodFill* znajdującą się w bibliotece OpenCV. Na końcu dokonywana jest filtracja medianowa (o elemencie strukturującym 5x5) za pomocą, której oczyszczane są brzegi elipsy z niepotrzebnych szumów widocznych na Rysunku 79. Dzięki tak uzyskanej masce możliwe jest wybranie części obrazu zawierającej obiekt, co ilustruje Rysunek 80. Widać na nim wybraną klatkę sekwencji, na której znajduje się oznaczona elipsą twarz, która jest w tym wypadku przykładowym obiektem. Poprzez stworzenie odpowiedniej maski z wybranego fragmentu obrazu został on właściwie wyselekcjonowany.

a)



b) c) d)



Rysunek 80. Przykład zastosowania eliptycznej maski
a) oznaczenie obiektu, b) obiekt z tłem, c) maska, d) uzyskany obiekt (zastosowanie maski c) na obrazie b))

Oczywiście rozwiązanie to nie zawsze się sprawdza tak jak dla prostokątnych obiektów. W takim wypadku narożniki obiektu zostają pominięte w procesie tworzenia modelu, co w pewnym stopniu zniekształca jego właściwy wybór. Jednak dla większości obiektów, które zostały użyte w badaniach zastosowanie eliptycznej maski poprawia dokładność opis obiektu.

5.3 Cechy obiektu

Następnym krokiem było stworzenie części zawierającej opis obiektu, a dokładniej procesu rozpoznania jego w kolejnych klatkach sekwencji obrazów. Do tego celu wykorzystane zostało pojęcie histogramów. Histogramy dobrze opisują kolorystykę obrazu (jego fragmentu), gdyż przedstawiają rozkład występowania poszczególnych poziomów jasności wybranych składowych w obrazie. Dzięki temu mogą zostać użyte w procesie rozpoznania obiektów w sekwencjach obrazów, jako ich cechy charakterystyczne.

5.3.1 Wykorzystanie histogramów 1D

Zaimplementowany proces lokalizacji składa się z kilku etapów: w pierwszej kolejności wyznaczane są histogramy każdej składowej RGB wzorca oraz kolejnych fragmentów aktualnego obrazu sekwencji (przy czym wymiary okna oraz modelu są identyczne), a w końcowym etapie dokonywane jest ich porównanie. Na podstawie prób określony został próg rozpoznania obiektu: 0.7 * wartości maksymalnego dopasowania (jest on wyznaczany poprzez analizę kilku kolejnych klatek sekwencji):

$$(comp_{rgb})_n = 0.7 * \max\,(comp_{rgb})_{n=0}^{n-1}$$

gdzie:

$comp_{rgb}$ – wartość średnia porównania;

n – kolejne obrazy sekwencji;

Przyjęto, iż wpływ każdej ze składowych RGB na rozpoznanie jest jednakowy, dlatego ostateczny wynik porównania otrzymywany jest na podstawie odpowiedniej średniej. Istotne jest wybranie fragmentu o jak największej wartości $comp_{rgb}$:

$$comp_{rgb} = \frac{(comp_r + comp_g + comp_b)}{3}$$

Wartość dopasowania każdej ze składowych RGB uzyskiwana jest poprzez porównanie badanego fragmentu z wzorcem według określonej metody porównawczej. Biblioteka OpenCV udostępnia kilka takich metod: [1]

- CV_COMP_CORREL – metoda ta porównuje histogramy zgodnie z poniższym wzorem;

$$d_{correl}(H_1, H_2) = \frac{\sum_i H_1'(i) \cdot H_2'(i)}{\sum_i H_1'^2(i) \cdot H_2'^2(i)}$$

gdzie:

H_1, H_2 – porównywane histogramy;

Natomiast H_1' iH_2' obliczane są z wzoru:

$$H_k'(i) = H_k(i) - \frac{1}{N}\left(\sum_J H_k(j)\right)$$

gdzie:

N – liczba przedziałów histogramu;

- CV_COMP_CHISQR – metoda ta porównuje histogramy zgodnie z poniższy wzorem;

$$d_{chi-square}(H_1, H_2) = \frac{(\sum_i H_1(i) - H_2(i))^2}{\sum_i H_1(i) + H_2(i)}$$

- CV_COMP_INTERSECT– metoda ta porównuje histogramy zgodnie z poniższym wzorem;

$$d_{intersection}(H_1, H_2) = \sum_i \min(H_1(i), H_2(i))$$

- CV_COMP_BHATTACHARYYA– metoda ta porównuje histogramy zgodnie z poniższym wzorem;

$$d_{Bhattacharyya}(H_1, H_2) = \sqrt{1 - \sum_i \frac{\sqrt{H_1(i) \cdot H_2(i)}}{\sqrt{\sum_i H_1(i) \cdot \sum_i H_2(i)}}}$$

- EMD – (ang. Earth Mover Distance) metoda ta porównuje oba histogramy poprzez ocenę ile należy wykonać operacji, aby z jednego histogramu stworzyć drugi;

Dla każdej z powyżej wymienionych metod wynikowa wartość porównania jest inna. Schemat przyjmowanych wartości przedstawia poniższa tabela:

Histogram		Metoda porównywania histogramów				
Model		Correlation	Chi square	Intersection	Bhattacharyya	EMD
Dokładne dopasowanie		1,00	0,00	1,00	0,00	0,00
Połowiczne dopasowania		0,70	0,67	0,50	0,55	0,50
Całkowite niedopasowanie		-1,00	2,00	0,00	1,00	1,00

Tabela 4. Metody porównywania histogramów zaimplementowane w bibliotece OpenCV

Istotne było to, aby wybrać optymalna metodę, na która mają najmniejszy wpływ zakłócenia, minimalne zmiany obiektu itd. Aby właściwie wybrać odpowiednią metodę każda z nich została sprawdzona na jednym obrazie w różnych skalach. Teoretycznie wyniki porównania powinny być bliskie dokładnemu dopasowaniu, mimo iż tak naprawdę zmniejsza się ilość pikseli w obrazie gdy ulega on zmniejszeniu. Poniżej zamieszczony został użyty obraz oraz wyniki badań:

Rysunek 81. Badany obraz w różnych skalach.

Skala [%*bok]	Metoda porównywania histogramów				
	Correlation	Chi square	Intersection	Bhattacharyya	EMD
100	1,00000	0,00000	1,00000	0,00000	0,00000
75	0,99934	0,00624	0,96905	0,04135	0,07666
50	0,99856	0,02465	0,94089	0,08546	0,15793
25	0,99664	0,05846	0,90645	0,14442	0,31037
10	0,98270	0,21758	0,79682	0,29097	0,71354

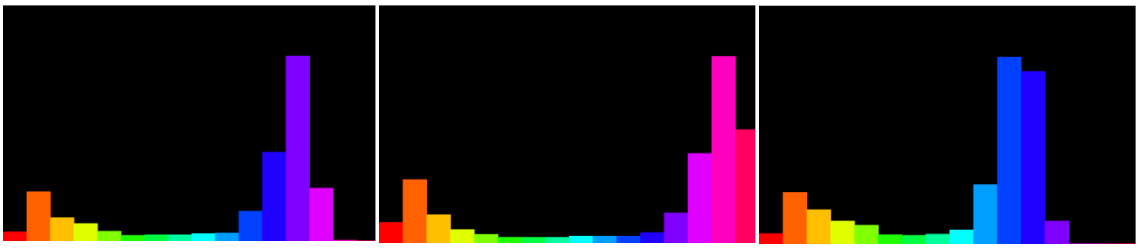
Tabela 5. Wyniki badań metod porównywania histogramów w zależności od zmiany skali obrazu

Na podstawie powyższych badań można zauważyć, że zmiana parametrów obrazu ma wpływ na wyniki porównania każdej z badanych metod. Jednak najlepsza z nich okazała się metoda korelacji CV_COMP_CORREL, której wyniki były w najmniejszym stopniu zależne od wprowadzanych zmian. Z tego powodu została ona wykorzystana podczas dalszych badań.

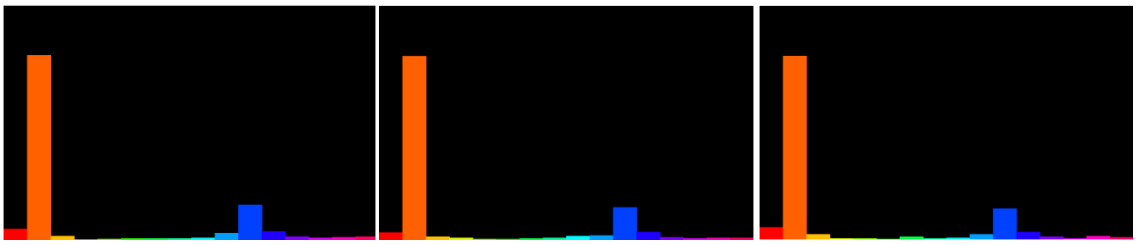
Niezależnie od stosowanej metody duży wpływ na wartości składowych RGB ma zmienne oświetlenie. Aby ograniczyć jego negatywne działanie zastosowana została konwersja z przestrzeni RGB do HSV. Wpływ oświetlenia na wartości poszczególnych składowych dobrze ilustruje Rysunek 82. Przedstawia on przykładowy obiekt dla różnych poziomów oświetlenia zarówno jaśniejszego jak i ciemniejszego z wykorzystaniem dwóch paletach kolorów: RGB i HSV (bez użycia eliptycznej maski). Dla przykładu umieszczony został histogram składowej B (Rysunek 82.b), na którym zauważyć można zmianę jej wartości zależnie od oświetlenia (składowa ta w dużej mierze zawiera opis skuwki będącej przykładowym obiektem)). Oprócz niego zamieszczone zostały histogramy każdej ze składowych HSV otrzymane z obrazu po konwersji RGB -> HSV.

a)

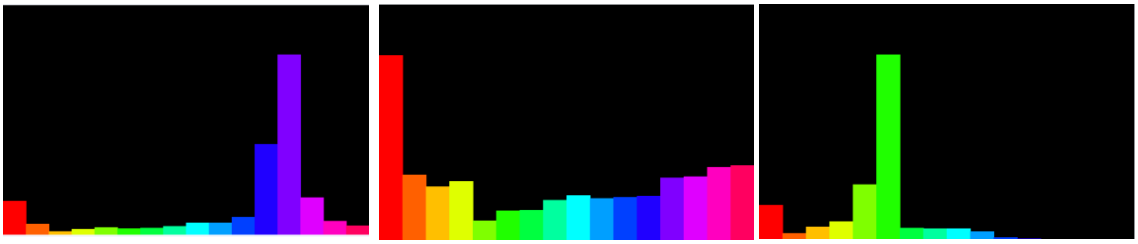
b)



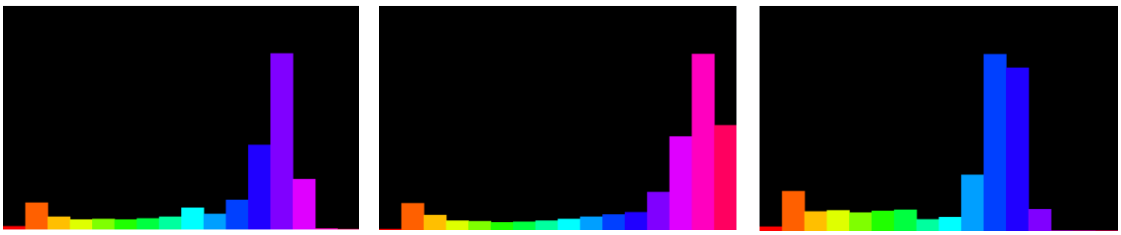
c)



d)



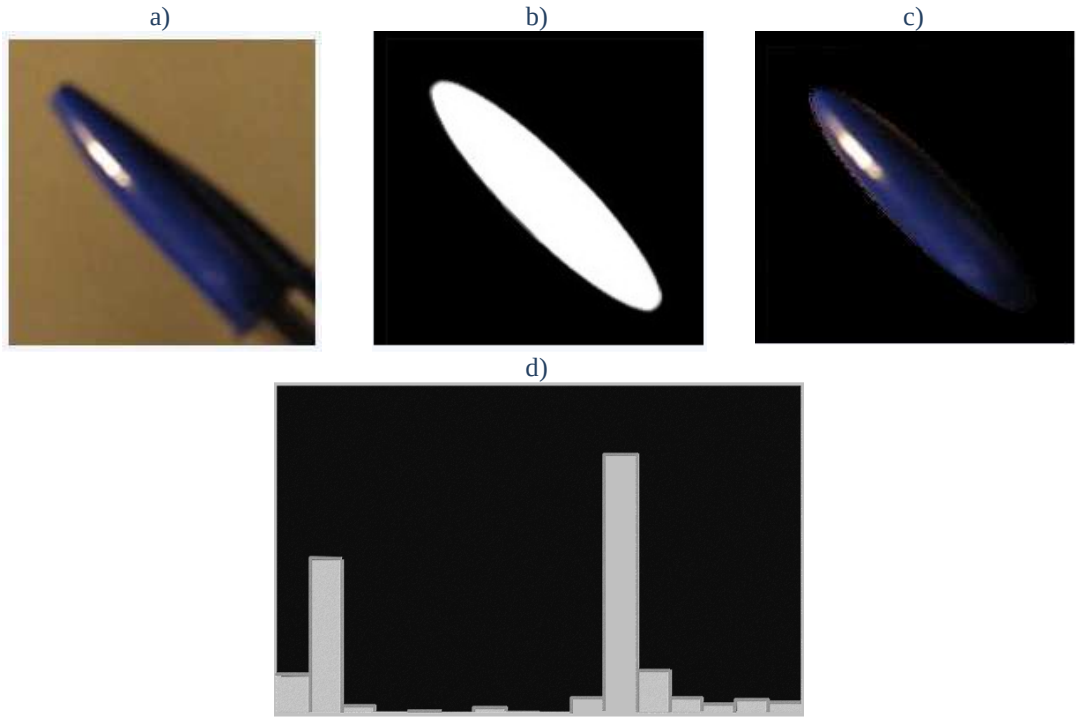
e)



Rysunek 82. Obraz zawierający obiekt przy zmiennym oświetleniu a) w przestrzeni RGB wraz z b) histogramem składowej B, oraz po konwersji obrazów z przestrzeni RGB do HSV c) histogramem składowej H, d) histogramem składowej S, e) histogramem składowej V

Po analizie Rysunku 82 stwierdzić można że zmienne warunki, którym został poddany przykładowy obraz zawierający badany obiekt mają najmniejszy wpływ na

składową H (Rysunek 82.d). Na jej rozkładzie zauważyć jednak można duży wpływ tła (kolorystyka mieszcząca się w pomarańczowym słupku). Jest to spowodowane tym, iż brany był pod uwagę cały prostokątny obraz zawierający model obiektu. W tym wypadku jak i wielu innych celowe było zastosowanie eliptycznej maski, dzięki której do rozpoznania wybierana jest jedynie część obrazu. Ilustruje to poniższy rysunek, na którym widać, że przy wyznaczaniu histogramu użyta została jedynie część obrazu zawierająca przykładową skuwkę.

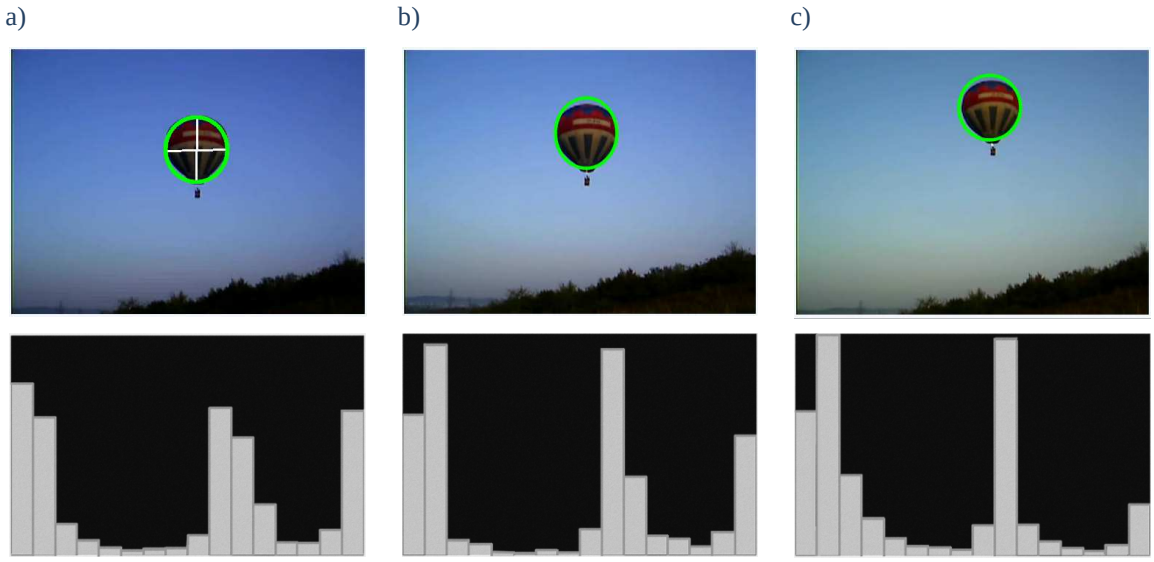


Rysunek 83. Ilustracja wpływu zastosowania eliptycznej maski na histogram składowej H: a) model, b) maska, c) wynik, d) histogram składowej H

Ze względu na to, iż zastosowanie maski jedynie ogranicza ilość pikseli wykorzystywaną do tworzenia histogramu (nie ma wpływu na zmianę składowych pod wpływem oświetlenia), również w tym przypadku można przyjąć, że składowa H ulega najmniejszym zmianom pod wpływem zmiennych warunków oświetleniowych. Fakt ten skłania do nadania jej większej wagi przy rozpoznawaniu obiektu, gdyż w ten sposób obniżany jest wpływ zakłóceń wywołanych rozjaśnieniem czy zacienieniem obrazu, które najczęściej występują w środowisku. Poniższe równanie przedstawia zależność rozpoznania od każdej ze składowych (przyjęte wartości zostały dobrane na podstawie kilkunastu prób):

$$comp_{hsv} = 0.7 * comp_h + 0.2 * comp_s + 0.1 * comp_v$$

Na Rysunku 84. zamieszczony został przykład wybranego wzorca oznaczony zieloną elipsą oraz przykładowa klatka ilustrującą jego rozpoznanie. Pod obrazami znajdują się rozkłady składowej H. Od razu zauważyć, że przykładowy obiekt, którym w tym przypadku był balon ma dużą złożoność kolorystyczną, lecz mimo tego, że zmieniły się warunki (dobrze widoczne po zmianie kolorów nieba) został właściwie rozpoznany.

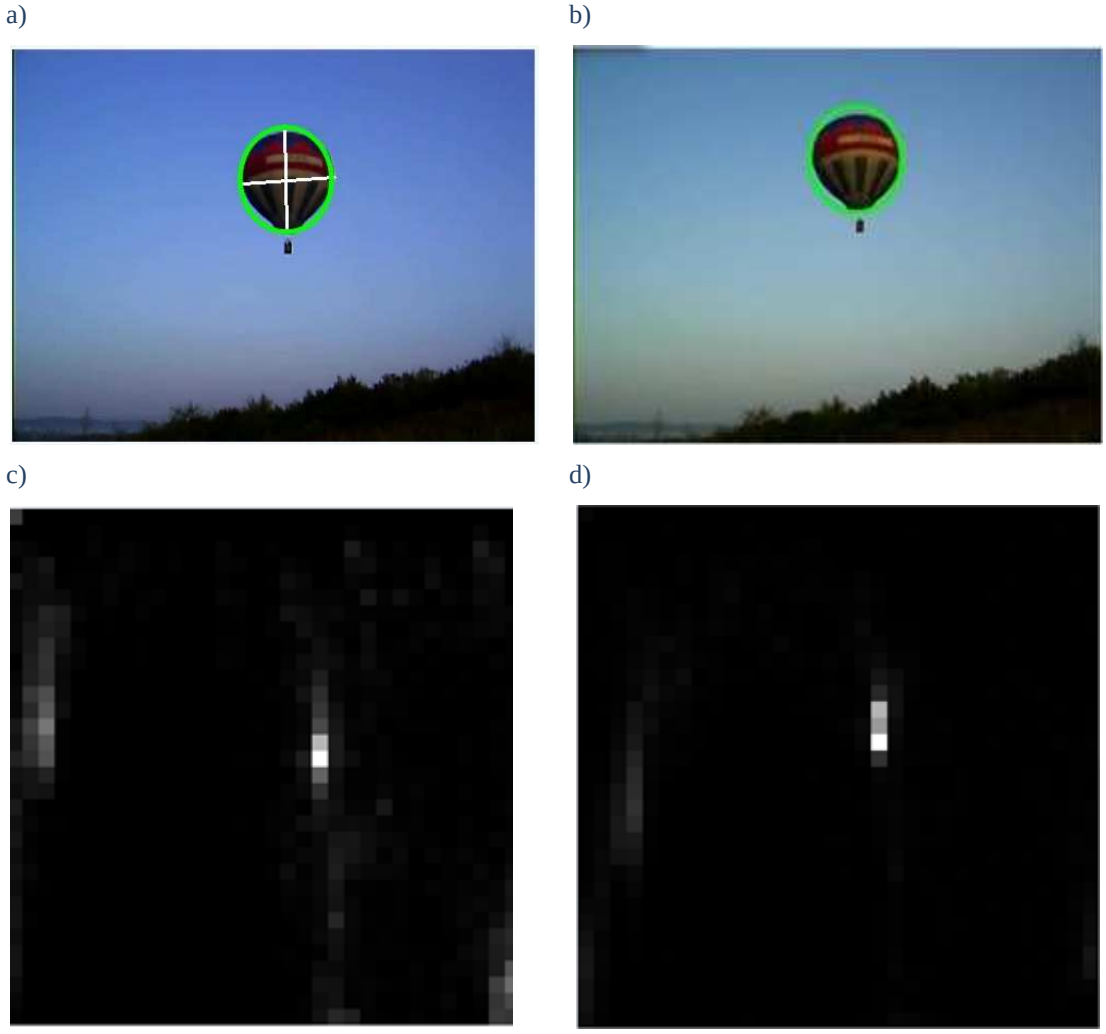


Rysunek 84. Przykład działania programu:
wzór obiektu, b), c) znaleziony obiekt,
oraz odpowiadające histogramy składowej H

5.3.2 Wykorzystanie histogramów 2D

Dotychczas przyjęte założenia odnoszące się do wpływu poszczególnych składowych na rozpoznanie obiektu w pewnym stopniu ograniczały wykorzystanie informacji zawartych w histogramach. Przyjęcie jednak zasady równego udziału w lokalizacji obiektu każdej ze składowych było niemożliwe gdyż ich wartości zmieniały się w różnym stopniu w zależności od warunków oświetlenia. Aby możliwe było wykorzystanie większej ilości informacji bez konieczności ich ważenia wprowadzone zostało pojęcie histogramów wielowymiarowych tak jak 2D oraz 3D. Polega ono na połączeniu danych zawartych w kilku składowych poprzez sprawdzanie nie tylko częstotliwości występowania poszczególnych barw, ale

również ich nasycenia czy jasności na jednym histogramie. Ze względu na to, iż składowa V jest najbardziej zmienna pod wpływem oświetlenia, które wprowadza najwięcej zakłóceń została wyeliminowana z procesu rozpoznania. Takie podejście ma również wpływ na przyśpieszenie działania algorytmu, gdyż program nie musi wykonywać dodatkowych operacji dla tej składowej. W ten sposób w algorytmie śledzenia zastosowane zostało porównywanie histogramów 2D opartych na składowych H i S.



Rysunek 85. Przykład a) modelu wraz z histogramem 2D składowych H i S oraz b) rozpoznanego obiektu wraz z histogramem 2D składowych H i S

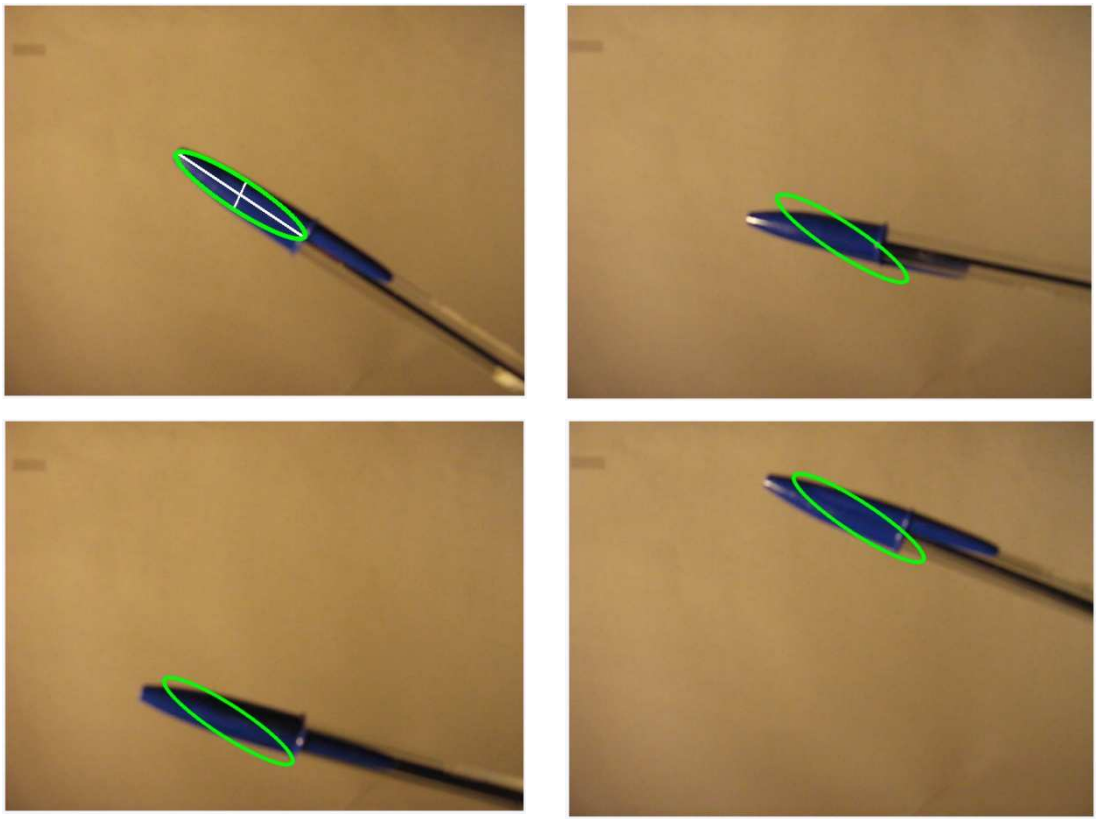
Rysunek 85 przedstawia przykładowe rozpoznanie obiektu poprzez wykorzystanie porównywania histogramów 2D. Obiekt został właściwie rozpoznany, mimo iż zmieniło się oświetlenie oraz nasycenie barw. Jak widać na obrazach histogramów w dużej mierze przyczynił się do tego oznaczony bielą obszar – kolor niebieski o odpowiednim nasyceniu (dużo większym niż nieba w tle obiektu), które jest bardzo zbliżony do wzorca.

5.4 Zmiany obiektu

Wiele obiektów podczas ruchu zmienia pozycję względem obserwatora tak jak obracają się czy oddalają itd., co powoduje, że w obrazie zmienia się ich orientacja czy wielkość. Ma to duży wpływ na wartości dopasowania, gdyż maska utworzona na podstawie modelu nie wybiera odpowiednich pikseli do prawidłowego rozpoznania (obiekt staje się za mały lub za duży lub też nieprawidłowo obrócony). W takim wypadku należało modyfikować maskę i sprawdzać, dla jakich jej parametrów można uzyskać najlepszy wynik rozpoznania. Aby wybrać optymalną maskę w całym zakresie możliwych parametrów trzeba by było wielokrotnie przeszukać cały obraz z wykorzystaniem różnych masek. Takie podejście było zbyt czasochłonne z powodu ogromnej liczby koniecznych do wykonania operacji i celowe było wprowadzenie jakichś ograniczeń. Pierwszym założeniem była stałość kształtu maski. Większość badanych obiektów tylko nieznacznie zmieniała kształt, a sprawdzanie jego zmienności jest zagadnieniem bardzo złożonym, wymagającym bardzo dużego czasu przetwarzania. Drugim założeniem było przyjęcie maksymalnej rotacji oraz oddalenia czy przybliżenia obiektu względem rejestratora pomiędzy kolejnymi klatkami sekwencji. Miało to swoją celowość ze względu na trudność w rozpoznaniu szybko zmieniających położenie obiektów, dla których występują charakterystyczne rozmycia obrazu. Poza tym większość badanych obiektów nie zmienia znacznie swojej pozycji względem rejestratora, dlatego można przyjąć odpowiednie ograniczenia :np. maksymalny obrót $\pm 10\,°$ oraz zwiększenie/zmniejszenie obiektu $\pm 10\,\%$. Zmniejszy to w dużym stopniu liczbę wykonywanych operacji, a co za tym idzie znacznie skróci czas przetwarzania, przy czym nie będzie miało znaczącego wpływu na zmniejszenie prawdopodobieństwa rozpoznania obiektu.

5.4.1 Obrót obiektu

Zmiana orientacji obiektu jest zjawiskiem często występującym w rzeczywistości. Każde przechylenie, wygięcie, obrót ma wpływ na właściwe rozpoznanie. Algorytm bez możliwości sprawdzania rotacji obiektu nie działa optymalnie, co ilustruje Rysunek 86.



Rysunek 86. Przykład śledzenia ze zmienną orientacją

Aby właściwie śledzić takie obiekty konieczne jest odpowiednie zmodyfikowanie maski, a dokładnie elipsy znajdującej się w niej w zależności od obrotu obiektu. Funkcja rysująca zawiera parametr umożliwiający wpisania odpowiedniego kąta położenia. Do tego celu wykorzystana została jednak macierz rotacji. Było to spowodowane przyszłościowym rozwojem algorytmu o sprawdzanie zmiany wielkości obiektu – jego oddalenie lub przybliżenie. Macierz ta daje prosty dostęp do skalowania obrazem, a co za tym idzie również maską dzięki której do porównania wybierana jest właściwa części fragmentu obrazu. Jej postać przedstawia poniższy wzór:

$$M_{3X3} = \begin{bmatrix} scale * \cos\omega & \text{scale} * \sin\omega & x \\ -\sin\omega & \cos\omega & y \\ 0 & 0 & 0 \end{bmatrix}$$

gdzie:

scale – współczynnik skali (scale = 1 – bez zmian);

ω – kąt obrotu;

x – współrzędna x punktu obrotu;

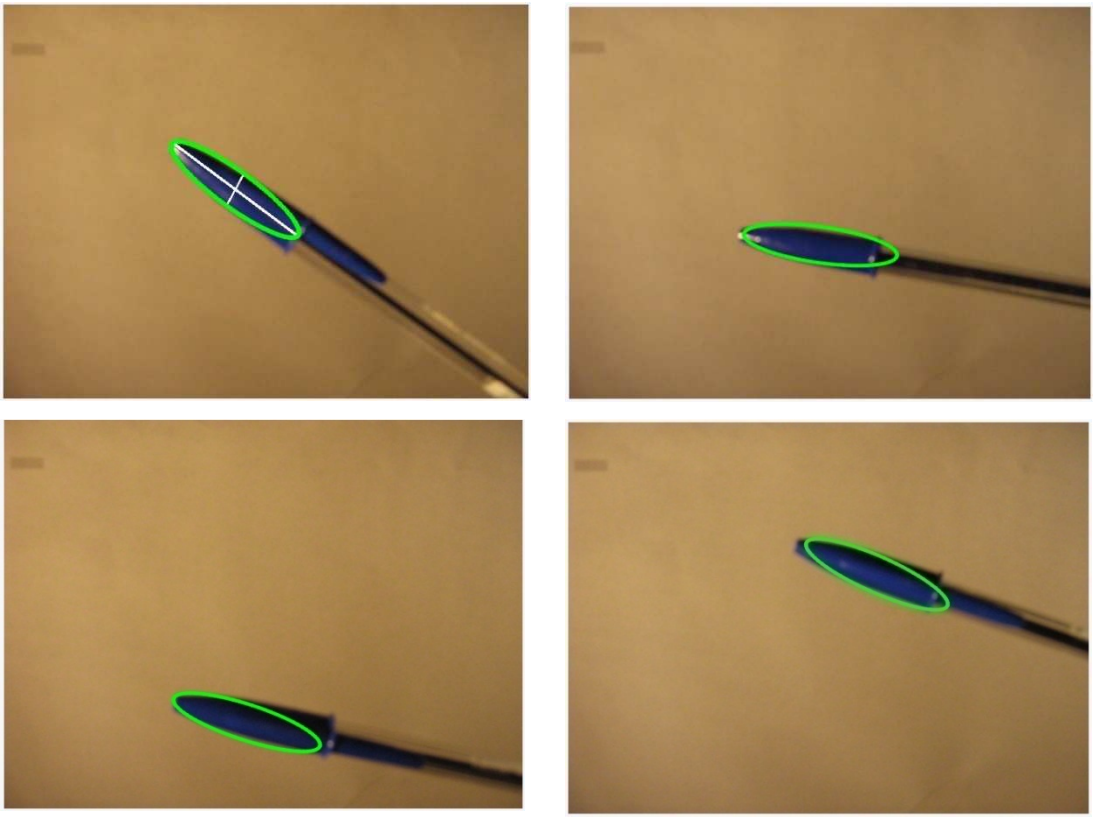
y – współrzędna y punktu obrotu;

Poniżej zamieściłem kilka przykładowych masek o różnych kątach obrotu:

a) b) c)

Rysunek 87. Przykładowe maski o różnych kątach obrotu a) 30 °, b) 50 °, c) 80 °

Rotacja sprawdzana jest w punkcie, w którym znalezione zostało maksymalne dopasowanie zarówno zwiekszając jak i zmniejszając kąt obrotu maski. Dla polepszenia możliwości algorytmu wprowadzona została wielostopniowa zmiana kąta uzależniona od wyników rozpoznania. Umożliwiło to dokładniejszą lokalizację zarówno szybko jak i wolno zmieniających orientację obiektów. Rotacja obiektu sprawdzana jest tylko wtedy, gdy jej wartość dopasowania jest większa od jego maksymalnej wartości dla danej klatki. Zastosowanie obrotowej maski zwiększa prawdopodobieństwo właściwego rozpoznania obiektu co bardzo dobrze ilustruje Rysunek 88. Przykład ten przedstawia śledzony obiekt, którym była skówka dla różnej orientacji w kolejnych klatkach sekwencji obrazów. Jak widać obrys zlokalizowanego obiektu jest lepiej dopasowany niż na Rysunku 86., co jest wynikiem zastosowania obrotowej maski.



Rysunek 88. Przykład śledzenia obiektów z wykorzystaniem maski elipsy oraz obrotu obiektu

5.4.2 Przybliżenie oraz oddalenie obiektu

Kolejną wprowadzoną modyfikacją był „zoom”. Przybliżanie, czy oddalanie obiektu względem obiektywu kamery to normalne zjawisko występujące w rzeczywistości. Ma to duży wpływ na wymiary obiektu, a co za tym idzie na prawidłowe rozpoznanie obiektu. Zastosowanie maski zgodnej z wzorcem w tym wypadku nie sprawdza się gdyż w pewnym momencie śledzony element zawiera się w części obszaru maski, a co za tym idzie do wyznaczania histogramu brane zostaje również tło. Adekwatnie, gdy obiekt się zwiększa maska wybiera jedynie jego część, co też może zakłócać właściwe rozpoznanie. Aby temu zapobiec wprowadzone zostało skalowanie z wykorzystaniem wcześniej przedstawionej macierzy rotacji. Mnożąc odpowiednie wartości tej macierzy przez współczynnik skali uzyskać można zmianę jej rozmiaru, co ilustruje Rysunek 89.

a) b) c)

Rysunek 89. Przykładowe maski o różnym skalowaniu: a) oryginał – współczynnik skali =1; b) zwiększenie współczynnika skali o 15 %; c) zmniejszenie współczynnika skali o 15 %

Jak widać na Rysunku 89 regulując współczynnik skali w prosty sposób zmieniana jest wielkość elipsy w masce. Umożliwiło to dopasowanie maski do rozmiarów obiektu, a co za tym idzie właściwsze porównanie histogramów z modelem. Oczywiście przeskalowywanie w pełnym zakresie byłoby zbyt czasochłonne i ze względów praktycznych bezcelowe maksymalny „zoom" pomiędzy klatkami został ograniczony. Przyjęte zostały następujące parametry: zmiana wymiarów, co 1-2 % do maksimum 3-6 %. Jeżeli pierwsze porównanie nie dało lepszego wyniku to dalsze etapy były pomijane.

5.5 Optymalizacja działania algorytmu

Optymalizacja w tym wypadku ma na celu poprawę działania algorytmu pod względem szybkości wykonywania oraz eliminację dotąd niezauważonych błędów. W pierwszej kolejności wprowadzane były proste modyfikacje stworzonego algorytmu zmniejszające ilość przeprowadzanych operacji. Dopiero w późniejszym etapie zostały one rozszerzone o bardziej złożone metody optymalizacyjne.

5.5.1 Zawężenie obszaru przeszukiwań

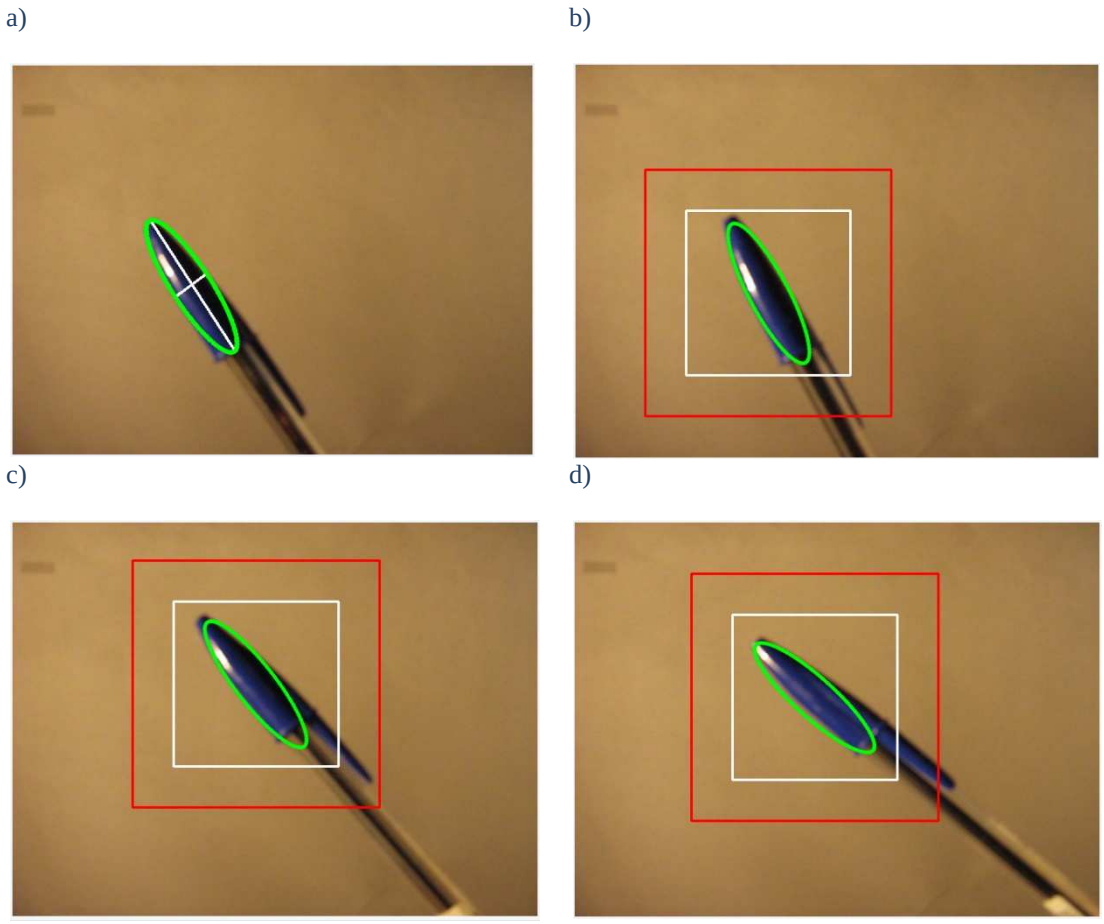
Przeszukiwanie pełnej klatki sekwencji obrazów w poszukiwaniu obiektu jest operacją bardzo czasochłonną i w dużej mierze zależy od rozdzielczości badanego obrazu. Na przykład dla sekwencji o wymiarach 600x400 i obiektu o wymiarach 100x100 potrzebne by było wykonanie 240000 – 10000 = 230000 porównań. Dodatkowo, jeśli każde porównanie to np. obliczenie sumy różnicy kwadratów dla wszystkich pikseli wzorca i fragmentu obrazu, ilość wykonywanych operacji robi się bardzo duża, co znacznie spowalnia proces lokalizacji obiektu. Ważne jest również to, że dla większości przypadków przesunięcie obiektu pomiędzy klatkami jest stosunkowo niewielkie, więc założone zostało, że do badań będzie używany jedynie wybrany fragment obrazu. Takie założenie potwierdza również fakt, że obiekty szybko się przemieszczające ulegają rozmyciu, występują gwałtowne ruchy kamerą, co znacząco utrudnia badania. Zjawisko to ilustruje zamieszony poniżej Rysunek 90, na którym widać przykład występowania charakterystycznej smugi zakłócającej właściwą lokalizację. Ze względu na to, iż problem ten jest bardzo złożony, a częstość występowania takich sekwencji nie jest duża ich analiza została pominięta podczas badań.

Rysunek 90 Przykład gwałtownego ruchu obiektu

Przy założeniu ograniczonego ruchu obiektu wprowadzone zostało zawężenie obszaru przeszukiwań. Polegało ono na zmniejszeniu okna, w którym poszukiwany był obiekt. W pierwszej kolejności zapisywany był punkt lokalizacji obiektu (dla pierwszej klatki ustalany podczas wyboru obiektu), a następnie na jego podstawie ograniczany był obszar poszukiwań, do którego brane pod uwagę było jedynie jego bliskie otoczenie. Proces ten ilustruje poniższy schemat (wartość 50 została przyjęta na podstawie kilkunastu prób):

1) przeszukanie całego obrazu w poszukiwaniu obiektu;
2) zapisanie pozycji (x_n,y_n) okna w którym został znaleziony;
3) w kolejnym obrazie sekwencji przeszukiwany jest obszar
   np. x_{n+1} ε (x_n-50, (x_n+50) + s_o), y_{n+1} ε (y_n-50, (y_n+50) + w_o),
   gdzie:
   s_o – szerokość okna, w_o – wysokość okna;
4) powrót do punktu 2;

Dzięki takiemu podejściu ograniczona zostaje ilość operacji porównywania. Przykład zastosowania zawężaniu obszaru przeszukiwań przedstawia poniższy Rysunek 91. Widać na nim oznaczony na zielono obiekt, wielkość okna, którym dokonywane jest przeszukiwanie oznaczony na biało oraz obszar przeszukiwań oznaczony na czerwono.



Rysunek 91. Przykład zawężania obszaru przeszukiwań: a) zaznaczony obiekt, b),c),d) ruch obiektu

Stała wielkość rozszerzenia przeszukiwanego obszaru ma kilka wad. Jedną z nich jest brak zależności jego rozmiarów od wartości dopasowania, co jest również zależne od szybkości ruchu obiektu. Gdy nastąpi duże przesunięcie obiektu może znaleźć się on np. na granicy okna przeszukiwań. W ten sposób obniża się maksymalna wartość dopasowania, jaką można uzyskać, przez co właściwa lokalizacja obiektu jest niemożliwa. Aby zapobiec takiej sytuacji wprowadzona została zależność wielkości obszaru przeszukiwań od wartości dopasowania – im większa jego wartość tym większy obszar i odwrotnie. Ilustruje to poniższy wzór:

$$var\_xy = var\_factor * (1 - comp_{hsv\_max})^3$$

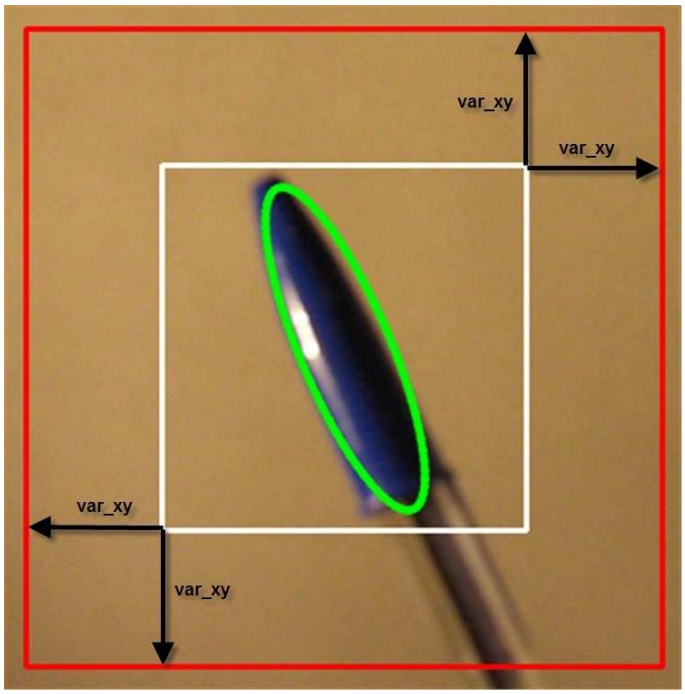
gdzie:

var_xy – wartość rozszerzania obszaru poszukiwań;

var_factor – współczynnik dla rozszerzania (przyjęta wartość: 250);

$comp_{hsv\_max}$ – maksymalna wartość dopasowania dla danej klatki;

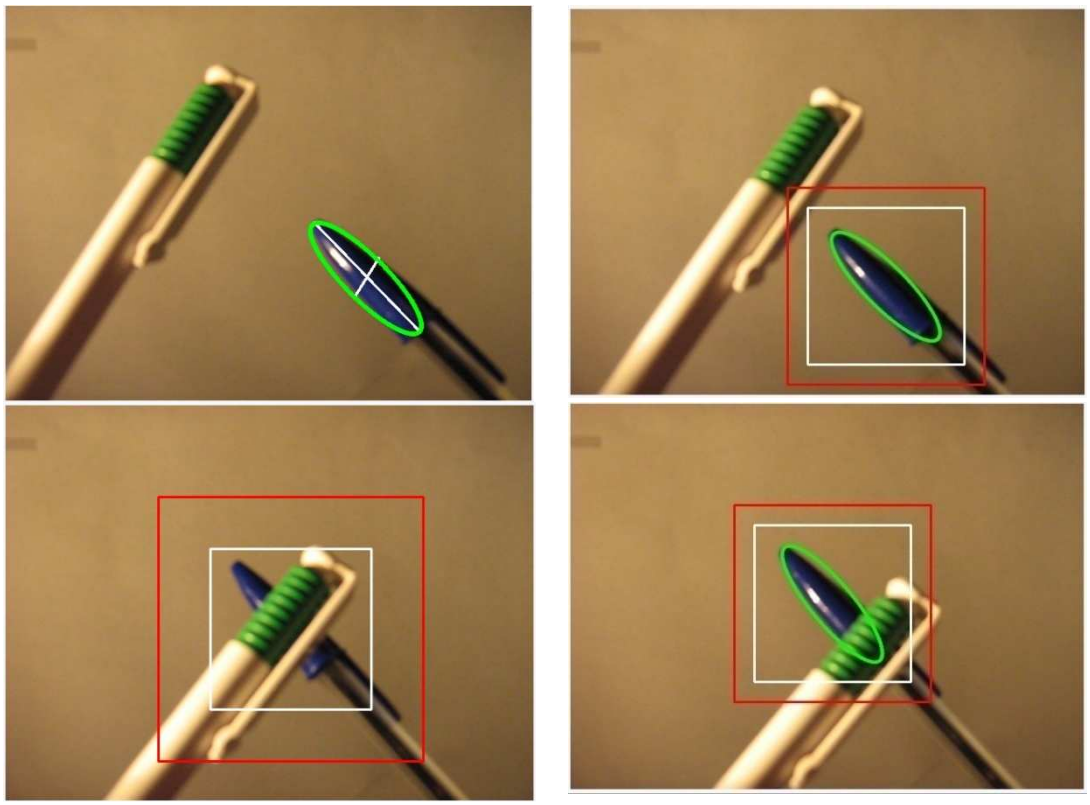
Wartość *var_xy* jest odpowiednio dodawana do współrzędnych punktu, w którym rozpoznany został obiekt tak, że wielkość obszaru zwiększa się o *2 x var_xy* co ilustruje poniższy rysunek.



Rysunek 92. Ilustracja wpływu wartości var_xy na rozszerzanie obszaru przeszukiwań

Ze względu na to, iż dla dużej wartości dopasowania ograniczenie obszaru przeszukiwań było bardzo małe np. dla dopasowania bliskiego 1 wartość *var_xy*=0 minimalna wartość rozszerzenia została przyjęta na poziomie 25. Zbyt mały obszar poszukiwań w dużym stopniu utrudniał wykrycie przesunięcia obiektu, co negatywnie wpływało na właściwe rozpoznanie.

Przykład zastosowania zmiennego obszaru przeszukiwań ilustruje Rysunek 93. Obiekt, którym w tym wypadku była niebieska skuwka został przysłonięty przez inny element. Jak można zauważyć w momencie, gdy śledzony obiekt został zasłonięty obszar jego poszukiwań został rozszerzony, natomiast przy właściwym rozpoznaniu został zawężony.



Rysunek 93. Przykład zmiennego obszaru poszukiwań

5.5.2 Zależność iteracji od wartości dopasowania

Poprzez zastosowanie zmiennych obszarów poszukiwań (często znacznie mniejszych niż obraz) zmniejszona została liczba wykonywanych operacji porównywania badanego fragmentu obrazu z wcześniej otrzymanym wzorcem. Mimo

to algorytm posiadał wady, które należało poprawić. Jedną z nich był sposób przeszukiwania wyznaczonego obszaru, który do tej pory był sprawdzany, co piksel w każdej z osi. Takie podejście dawało dużą dokładność otrzymanych wyników, ale było bardzo czasochłonne. Dla niskich wartości dopasowania (dużo mniejszych niż 1) bezcelowe było tak precyzyjne przeszukiwanie, gdyż prawdopodobieństwo tego, że obiekt znajdował się niedaleko było znikome. Odwrotną sytuację przedstawiały obszary o dużej wartości dopasowania, dla nich skok o 1 piksel pomiędzy kolejnymi iteracjami był bardziej uzasadniony. Aby właściwie obsłużyć te dwa wypadki wprowadzony został zmienny ruch okna zależny od tego, w jakim stopniu badany fragment jest zbliżony do modelu. Przyjęty sposób wyznaczania „skoku" okna pomiędzy iteracjami ilustruje poniższy wzór:

$$jump = jump\_factor * \sqrt[2]{(\,1 - \ comp_{hsv}\,)}$$

gdzie:

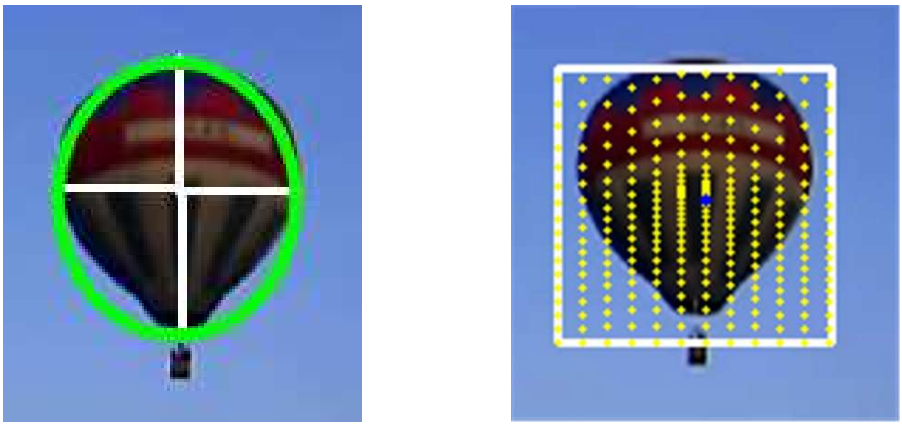
jump – wartość skoku;

jump_factor – współczynnik skoku (przyjęta wartość 10);

$comp_{hsv}$ – bieżąca wartość dopasowania;

Dzięki zastosowaniu zmiennego skoku omijana jest część wcześniej porównywanych fragmentów obrazu np. dla wartości dopasowania wynoszącej 0.9 przeskok wynosi 1, dla 0,5 wynosi 7 itd. Oczywiście możliwa jest odpowiednia regulacja, lecz trzeba pamiętać o tym, aby zbytnio nie zwiększyć wartości skoku, gdyż obiekt może zostać pominięty w czasie dokonywania porównań.

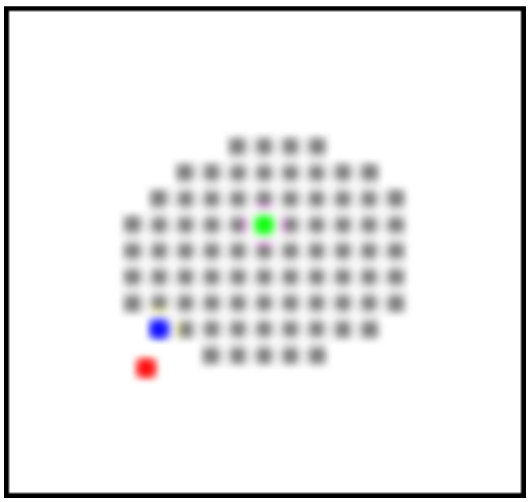
a) b)



Rysunek 94. Przykład zastosowania zmiennego skoku: a) zaznaczony obiekt, b) punkty zmiennego skoku

5.5.3 Likwidacja błędnych skoków w lokalizacji obiektu

Podczas testów zauważony został błąd w lokalizacji obiektu polegający na zbyt dużym przemieszczeniu okna rozpoznania w stosunku do przemieszczenia obiektu. Wynikało to z tego, że w okolicach punktu maksymalnego dopasowania często znajdują się obszary o podobnej wartości rozpoznania natomiast niższe od wybranego. Dany wynik działania algorytmu jest często obarczony błędem wywołanym szumami powodującymi niedokładną lokalizacje obiektu. Aby zapobiec zbyt dużym skokom w przemieszczeniu obiektu pomiędzy klatkami wprowadzone zostały następujące poprawki: pierwszym pomysłem był wybór punktu z określonego obszaru, który był najbliższy punktowi maksymalnego dopasowania z poprzedniej klatki sekwencji. Polegało to na stworzeniu obrazu zawierającego wartości dopasowania wymnożone przez liczbę 255 określającą wyniki w skali szarości, a następnie odpowiedniemu przeszukaniu tego obrazu pod względem odległości od ostatniej lokalizacji. Do porównania brane były jedynie te piksele, których wartość była mniejsza od maksimum o 5 %, gdyż nie zawężając sprawdzania algorytm wracałby w ten sam punkt. Działanie minimalizacji ilustruje poniższy rysunek, na którym szare punkty przedstawiają wartości dopasowania powyżej 95 % maksymalnej uzyskanej wartości oznaczonej na zielono. Punkt czerwony oznacza punkt rozpoznania obiektu na poprzedniej klatce, natomiast najbliższym jemu punktem z określonego szarego obszaru jest punkt niebieski, który zostaje przyjęty, jako punkt lokalizacji obiektu.



Rysunek 95. Ilustracja działania minimalizacji odległości

Wybór minimalnej odległości w pewnym stopniu zapobiegał powstawaniu gwałtownych ruchów pomiędzy wynikiem rozpoznania w kolejnych klatkach sekwencji, lecz był czasochłonny. Wartości dopasowania dla danego fragmentu obrazu były sprawdzane dwa razy, w pierwszym etapie, gdy tworzony był ich obraz, a następnie gdy był on przeszukiwany pod względem odległości od punktu maksymalnego dopasowania z poprzedniej klatki. Aby przyspieszyć działanie algorytmu dwa kroki trzeba było połączyć w jeden. I w ten sposób powstał drugi pomysł, a mianowicie ważenie dopasowania w zależności od odległości od punktu rozpoznania z poprzedniej klatki. Polegało to na mnożeniu wyniku porównania wzorca z badanym fragmentem przez odpowiedni współczynnik zgodny z poniższym wzorem:

$$factor = \left(1 - wsp * \frac{distance}{0.5 * size}\right)$$

gdzie:

factor – współczynnik wagi dopasowania;

wsp – współczynnik wpływu wagi (w badaniach przyjęty jako 0.05);

distance – odległość bieżącego środka rozpoznawanego fragmentu od punktu, w którym został zlokalizowany obiekt w poprzedniej klatce;

size – wielkość przeszukiwanego okna;

Poprzez wyliczenie współczynnika wagi ostateczna wartość dopasowania dla danego fragmentu wynosił:

$$comp_{hsv} = factor * comp_{hsvtemp}$$

gdzie:

$comp_{hsvtemp}$– wartość dopasowania;

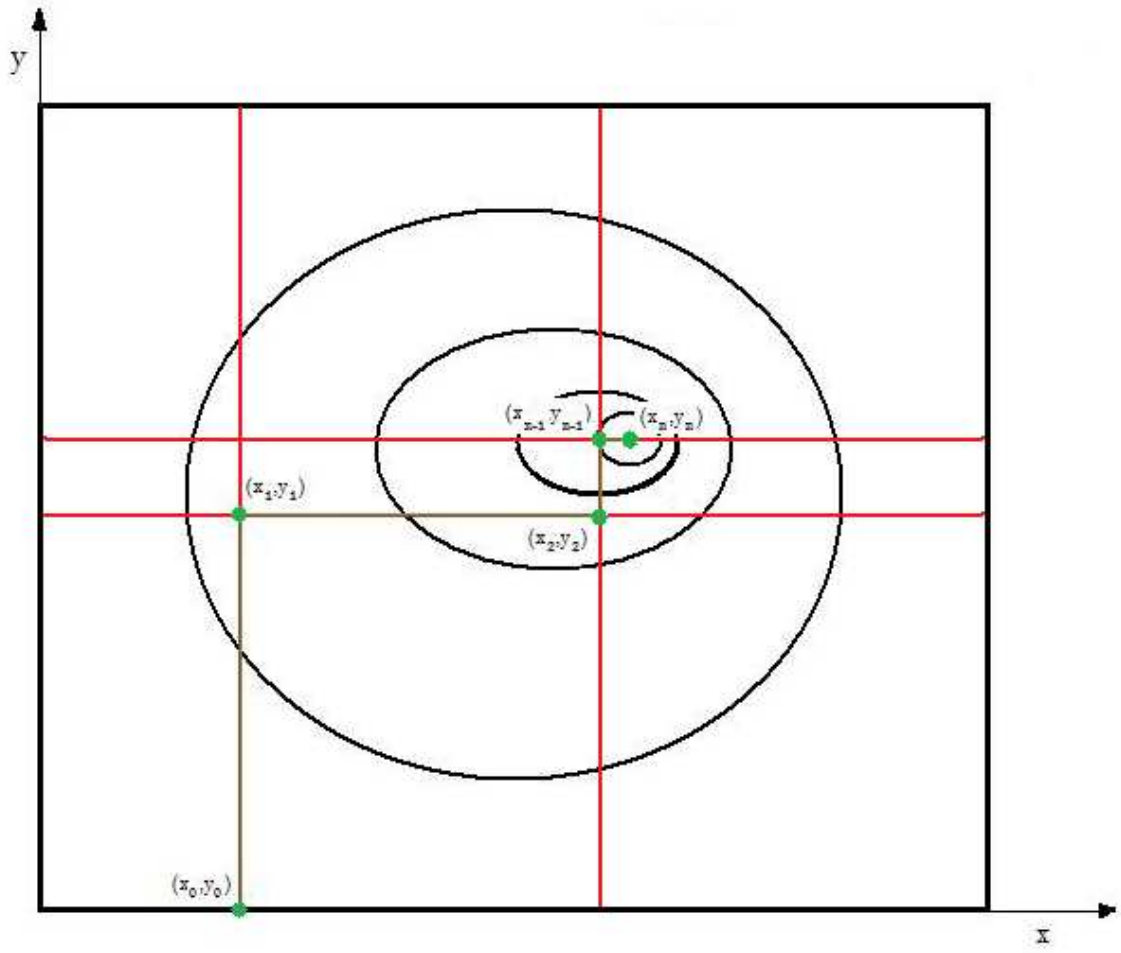
$comp_{hsv}$ – wynikowa wartość dopasowania;

Maksymalna wartość współczynnika wagi wynosi 1, z czego wynika, że dla większości przypadków obniża on wartość dopasowania, a w szczególności dla punktów odległych od punktu ostatniego dopasowania. W ten sposób wygładzane są wartości dopasowania, a co za tym idzie zmniejszony został wpływ „pików” – przypadkowych niewielkich obszarów o podwyższonym dopasowaniu.

Dzięki takiemu podejściu w dużym stopniu ograniczona została ilość wykonywanych operacji w stosunku do poprzedniego pomysłu bez zmniejszenia efektywności jego działania.

5.5.4 Skrócenie procesu lokalizacji z wykorzystaniem metody Gaussa-Seidla

Metoda Gaussa Seidla jest rozwinięciem metody spadku względem współrzędnych. W głównej mierze polega ona na minimalizacji (ewentualnie maksymalizacji) funkcji wzdłuż obu kierunków badanej bazy ortogonalnej B_1 … B_n. W przypadku zagadnień przetwarzania obrazów taką bazą może być obraz zawierający piksele w układzie kartezjańskim czy w odniesieniu do śledzenia obiektów wyniki dopasowania bieżącego fragmentu do modelu. Wyznaczenie maksimum polega na poszukiwaniu na zmianę maksimum wzdłuż każdej z osi, przy czym punktem startowym do kolejnej iteracji jest wynik poprzedniej. Metoda ta różni się od metody spadku względem współrzędnych skokiem, przy czym w metodzie Gaussa Seidla skok jest zmienny. Zasadę działania opisanej metody przedstawia poniższy Rysunek 96. W pierwszej kolejności poszukiwane jest maksimum w osi y, a następnie w osi x. Każde kolejne poszukiwanie zaczyna się z nowego punktu startowego tak jak w 1 iteracji z (x_0,y_0), w drugiej z (x_1,y_1) itd. aż do uzyskania maksimum lub przekroczenia założonego progu w punkcie (x_n, y_n) (punkty maksimum dla danej osi oznaczone na zielono). Czerwonymi liniami oznaczone zostały punkty przeszukane przez algorytm natomiast na zielono droga, którą się poruszał aby znaleźć maksimum. [36]



Rysunek 96. Ilustracja działania metody Gaussa Seidla

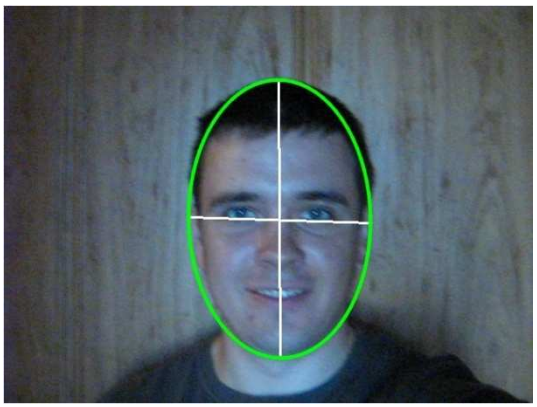
Jak od razu widać na powyższym rysunku dzięki zastosowaniu metody Gaussa Seida w znacznym stopniu zmniejsza się ilość wykonywanych operacji. Nie stosując tego typu optymalizacji przeszukiwany jest cały obraz czy jego ograniczony fragment tak jak w rozdziale 5.5.1. Również zmienny skoku pomiędzy kolejnymi porównaniami (rozdział 5.5.2) wpływa pozytywnie na szybkość działania algorytmu, ale dopiero połączenie tych wszystkich elementów z taką metodą jak metoda Gaussa Seida daje znaczące efekty obniżenia czasu działa programu. Zależność czasu lokalizacji obiektu w zależności od zastosowania różnych zestawów optymalizujących przedstawia poniższa tabela. Wynika z niej, że duży wpływ na szybkość rozpoznania obiektu ma zawężenie okna przeszukiwań, a najlepsze efekty można uzyskać stosując wszystkie zastosowane metody jednocześnie. Wyniki zostały uzyskane na podstawie średniej z 100 operacji rozpoznania dla obiektu o wymiarach 150x200 pikseli.

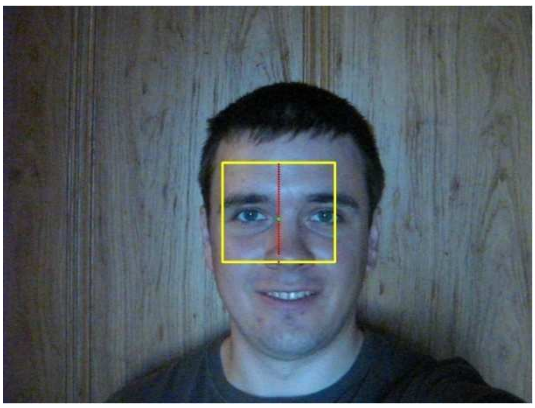
Optymalizacja	Czas lokalizacji [ms]
Metoda Gaussa Seidla	314
Metoda Gaussa Seidla + Zmienny skok	182
Metoda Gaussa Seidla + Zawężenie okna przeszukiwań	51
Metoda Gaussa Seidla + Zawężanie okna przeszukiwań + Zmienny skok	29

Tabela 6. Zależność czasu lokalizacji od zastosowanych metod optymalizacyjnych

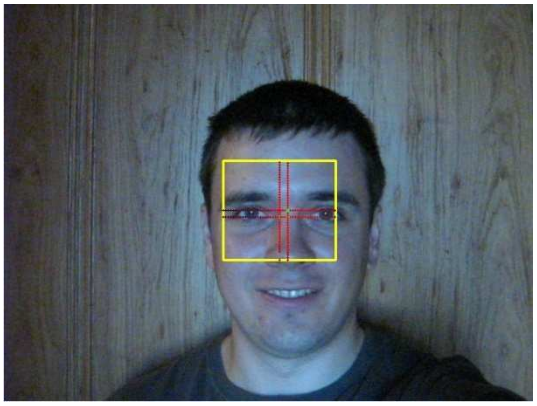
Przykładem zastosowania metody Gaussa Seidla jest Rysunek 97. Przedstawia on badany obiekt a), kolejne iteracje potrzebne do jego właściwej lokalizacji w następnej klatce sekwencji b) – f) oraz zlokalizowany obiekt g). Dodatkowo na czerwono zostały oznaczone linie siatki przeszukiwania metody Gaussa Seida, zielone punkty przedstawiające maksima w danej iteracji oraz żółty prostokąt obszar przeszukiwań. Badanym obiektem była twarz o wymiarach 215x330 pikseli, natomiast okno przeszukiwań ustaliło się na rozmiarze 137x116. Punktem startowym algorytmu jest punkt lokalizacji z ostatniej klatki ($x_{lastmax}$, $y_{lastmax}$), przy czym w pierwszej kolejności przeszukiwane są punkty wzdłuż osi y. Polega to na tym, że dla pierwszej iteracji x zmieniany jest z zakresu mieszczącego się w oknie przeszukiwań natomiast y jest stałe i wynosi y_{max}.

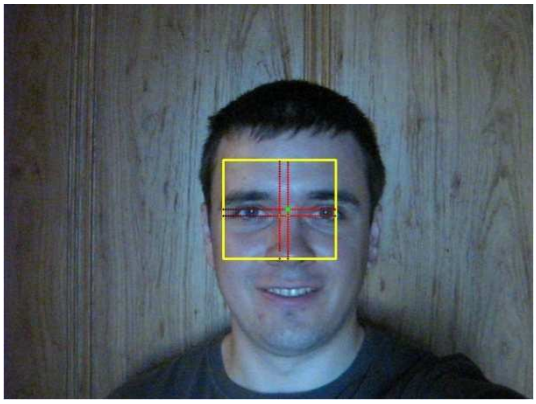
a) b)



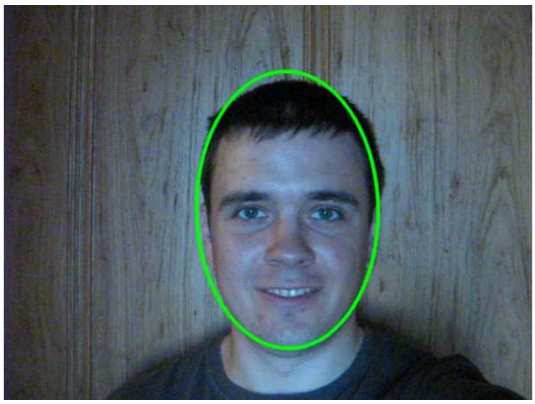


e) f)





g)



Rysunek 97. Przykład działania metody Gaussa Seidla zaimplementowanej w programie: a) zaznaczony obiekt, b - g) kolejne iteracje potrzebne do jego właściwej lokalizacji w następnej klatce sekwencji, g) zlokalizowany obiekt;

Jak widać na powyższym rysunku algorytm dobrze zlokalizował obiekt mimo stosunkowo niewielkiej liczbie 253 operacji porównywania badanych fragmentów obrazu z wzorcem. Oczywiście przyspieszenie działania algorytmu ponosi za sobą pewne konsekwencje: algorytm może ominąć rzeczywiste maksimum ze względu na jego złożone położenie (np. znajduje się pomiędzy innymi lokalnymi maksymami). Aby mieć większą pewność, że algorytm właściwie zlokalizuje badany obiekt

zaimplementowana została metoda Gaussa Seidla rozpoczynająca poszukiwania z dwóch punktów. Najpierw program znajduje pierwsze maksimum, a następnie drugie, przy czym jako ostateczny wynik brane jest to, którego wartość dopasowania jest większa. Dla pierwszego przypadku punktem startowym jest lewy dolny rób, a dla drugiego prawy góry róg okna przeszukiwań. W ten sposób zwiększa się czas lokalizacji obiektu, lecz dzięki temu istnieje większa pewność właściwej lokalizacji obiektu. Dla tego przypadku, jako punkty startowe zostały przyjęte punkty:

$$x_0 = x_{lastmax} - window_{width}/4; \;\; y_0 = y_{lastmax} - window_{height}/4 \; ;$$

oraz

$$x_0 = x_{lastmax} + window_{width}/4; \;\; y_0 = y_{lastmax} + window_{height}/4 \; ;$$

gdzie:
(x_0,y_0) – punkt startowy dla metody Gaussa Seidla;
$(x_{lastmax}, y_{lastmax})$ – punkt lokalizacji obiektu z ostatniej klatki;
$window_{width}$ – szerokość okna przeszukiwań;
$window_{height}$ – wysokość okna przeszukiwań;

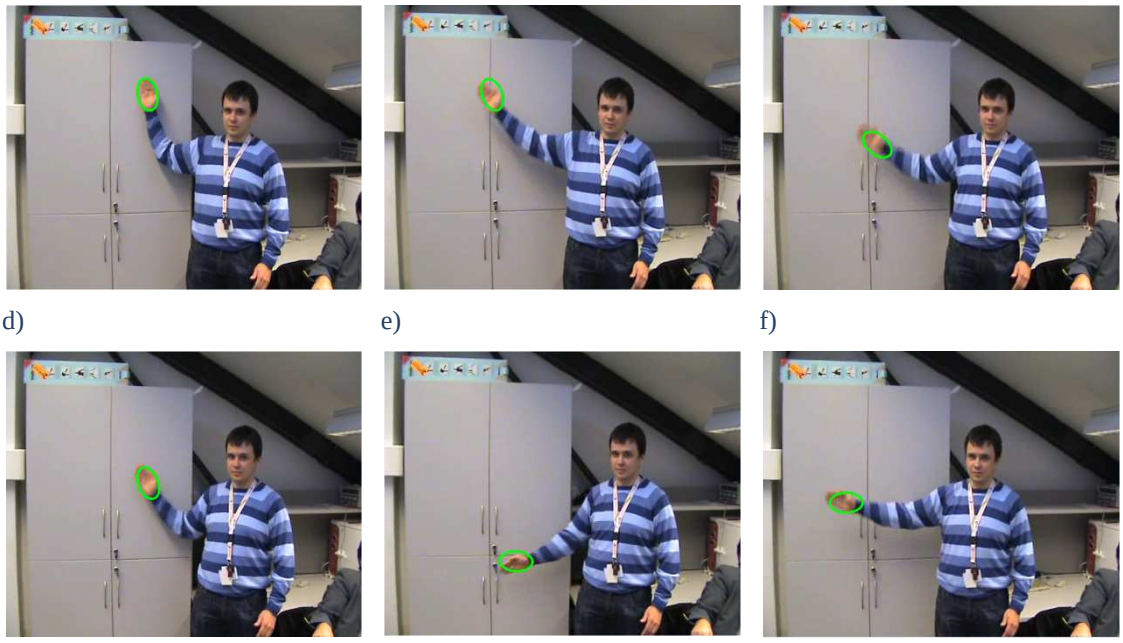
Punkty startowe zostały przyjęte ze względów praktycznych. Wybór punktów bliskich brzegom obszaru przeszukiwań dla większości obiektów nie sprawdzania się ze względu na ich stosunkowo niewielkie przemieszczenia, natomiast przyjęcie ich blisko punktu ostatniej lokalizacji jest bezcelowe, gdyż dla obu prób lokalizacji uzyskuje się zbliżone informacje. Działanie algorytmu jest zbliżone do Rysunku 96. Z tą różnicą, że lokalizacja odbywa się z dwóch punktów początkowych. Dla idealnego rozpoznania obie próby powinny dać identyczny wynik, lecz jest to trudne do uzyskania ze względu na różnego rodzaju zakłócenia, czy przyjęte uproszczenia tj. zmienny skok. W takim wypadku ważniejszy jest ten rezultat, który dał lepsze dopasowanie i on jest uznawany z punkt lokalizacji obiektu.

5.6 Wyniki badań

Na podstawie operacji opisanych w rozdziałach 5.1 - 5.5 powstał algorytm umożliwiający śledzenie obiektów na podstawie ich histogramów, wykorzystujący eliptycznie zaznaczanie obiektu. Oprócz tego zastosowane zostały różne techniki przyśpieszające jego działanie oraz umożliwiające wykrycie jego obrotu czy oddalenia/przybliżenia w stosunku do urządzenia rejestrującego. Aby ocenić jego skuteczność oraz szybkość jego działania został on poddany takim samym badaniom jak inne algorytmy opisane w rozdziale 4.

Poniżej przedstawione zostało działanie stworzonego algorytmu dla kilku wybranych sekwencji wykorzystywanych podczas badań:

a) b) c)



d) e) f)

Rysunek 98. Wyniki śledzenia przy użyciu stworzonego algorytmu dla Sekwencji 1

a) b) c)

d) e) f)

Rysunek 99.Wyniki śledzenia przy użyciu stworzonego algorytmu dla Sekwencji 3

a) b) c)

d) e) f)

Rysunek 100.Wyniki śledzenia przy użyciu stworzonego algorytmu dla Sekwencji 6

a) b) c)

d) e) f)

Rysunek 101.Wyniki śledzenia przy użyciu stworzonego algorytmu dla Sekwencji 8

a) b) c)

d) e) f)

Rysunek 102. Wyniki śledzenia przy użyciu stworzonego algorytmu” dla Sekwencji 9

a) b) c)

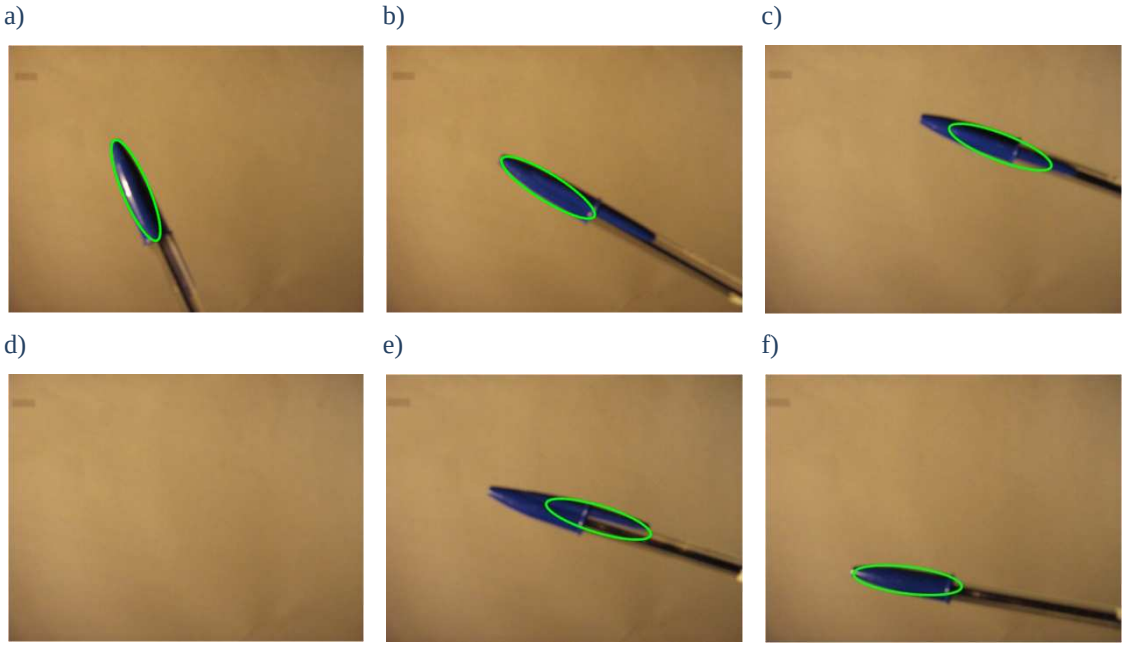
d e) f)

Rysunek 103. Wyniki śledzenia przy użyciu stworzonego algorytmu dla Sekwencji 11

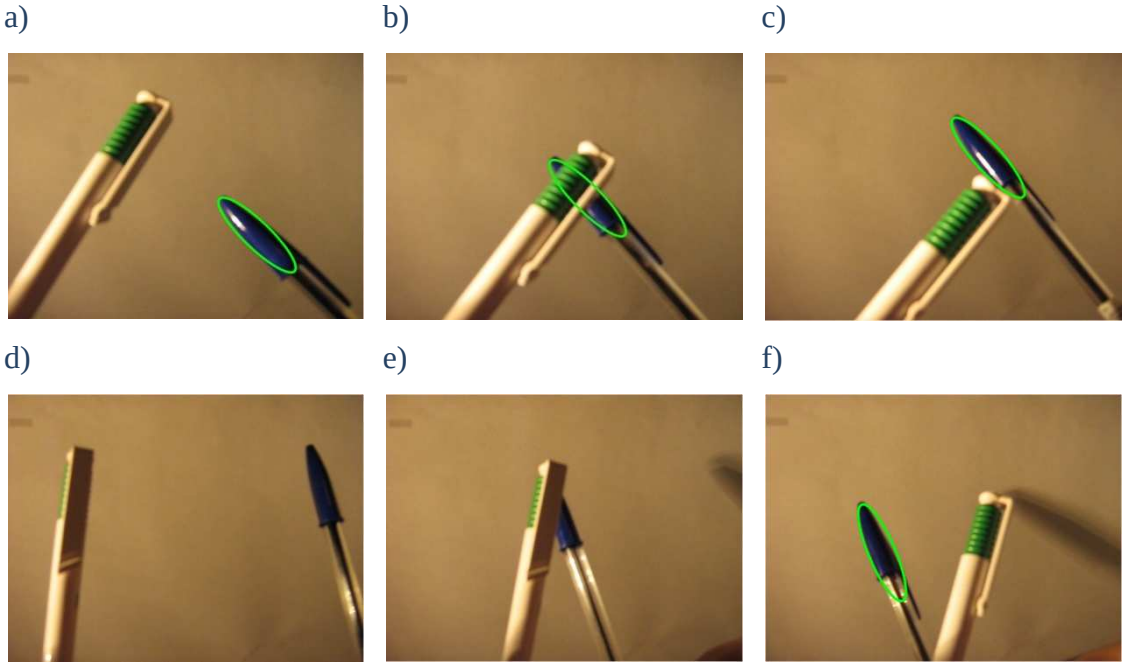
a) b) c)

d) e) f)

Rysunek 104. Wyniki śledzenia przy użyciu stworzonego algorytmu dla Sekwencji 12



Rysunek 105. Wyniki śledzenia przy użyciu stworzonego algorytmu dla Sekwencji 14



Rysunek 106. Wyniki śledzenia przy użyciu stworzonego algorytmu dla Sekwencji 15

Wyniki badań śledzenia obiektów:

Nr. sekwencji	Średni czas rozpoznania [ms]	Skuteczność działania [%]
1	17	88
2	20	100
3	22	92
4	13	100
5	11	69
6	13	7
7	7	4
8	9	86
9	49	78
10	10	43
11	65	57
12	76	91
13	28	45
14	63	69
15	88	89

Tabela 7. Wyniki badań śledzenia obiektów dla stworzonego algorytmu

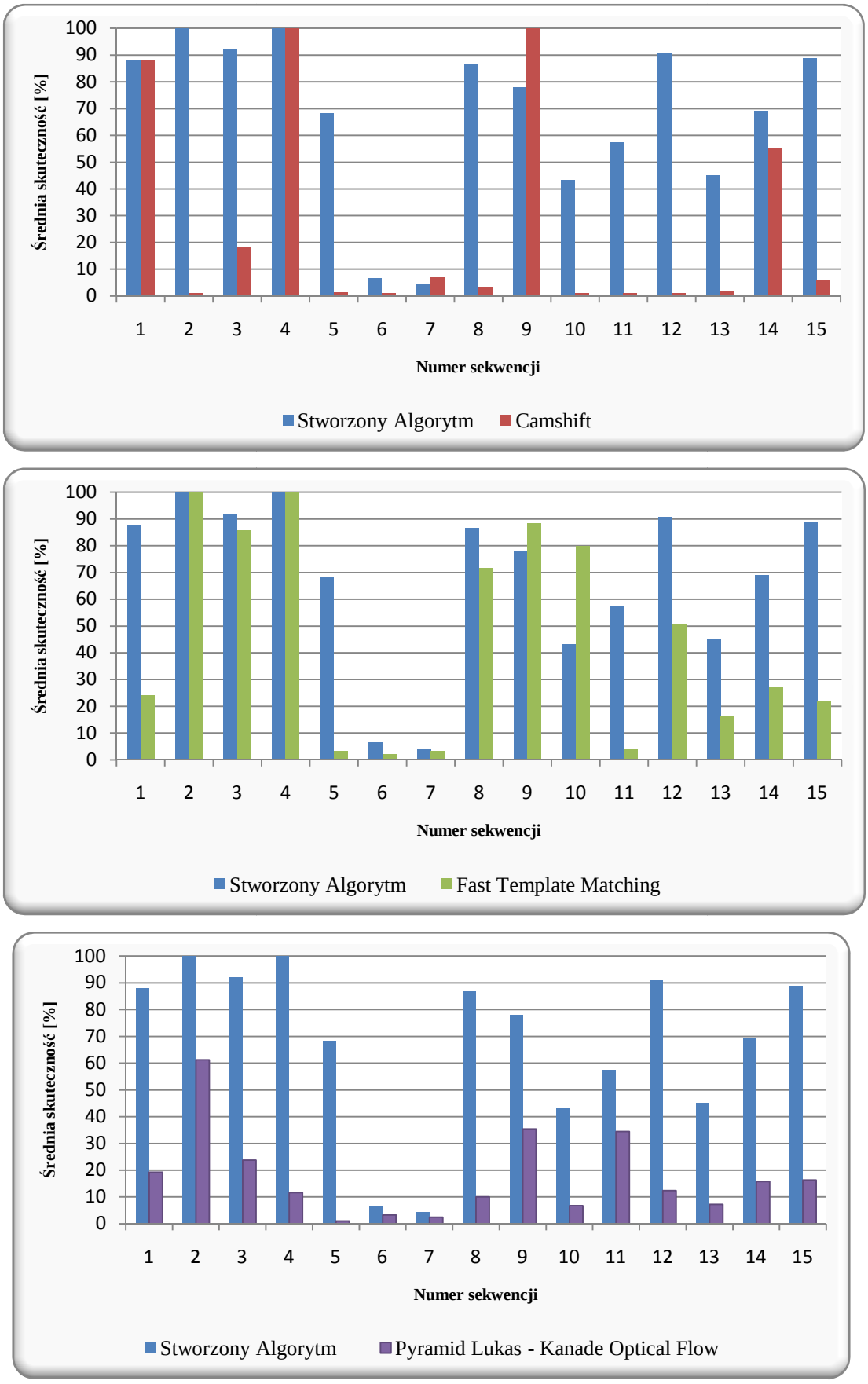
Na podstawie powyższych badań zauważyć można stosunkowo dużą skuteczność algorytmu dla większości badanych sekwencji obrazów (średnia z wszystkich wyników na poziomie 68 %). Szczególnie dobre wyniki uzyskane zostały dla sekwencji, w których obiekt był charakterystyczny w stosunku do tła i niewielkim stopniu zaszumiony (np. dla sekwencji 4 czy sekwencji 12 (Rysunek 104)). W tym wypadku histogramy zarówno modelu jak i porównywanych z nim fragmentów obrazów nie zawierały dodatkowych błędnych wartości, co znacznie ułatwiło rozpoznanie obiektu. Dla sekwencji o dużym wpływie czynników zewnętrznych takich jak słabe oświetlenie (sekwencja 7 (Rysunek 100)), czy gwałtowne ruchy obiektu (sekwencja 8) skuteczność algorytmu była słaba. Algorytm zamiast lokalizować badany obiekt przeskakiwał w różne fragmenty badanej sekwencji obrazów niewłaściwie lokalizując obiekt, co przedstawia Rysunek 100. Jedynym sposobem, aby temu zapowiedz jest przygotowanie lepszego oświetlenia oraz optymalnej sekwencji obrazów zawierającej rozpoznawany obiekt, co nie zawsze jest możliwe.

W powyższym rozdziale badane również było działanie algorytmu dla sekwencji obrazów, w których obiekt ulega obrotowi czy oddaleniu / przybliżeniu tak

jak dla sekwencji 12 (Rysunek 104), czy 14 (Rysunek 105). Takie zmiany położenia obiektu pomiędzy klatkami były stosunkowo niewielkie, co zwiększyło prawdopodobieństwo właściwej lokalizacji. Oczywiście nie zawsze rozpoznanie jest idealne, co widać na Rysunek 105.c i Rysunek 105.e, gdzie okno obrysowujące obiekt nieznacznie przesunęło się względem właściwej jego pozycji. Było to spowodowane tym, ze algorytm nie nadążał nad szybkościami zmian jego położenia. Jest to również widoczne na Rysunek 103, gdzie badany model oprócz wcześniej wspomnianej rotacji zmieniał kształt. Stworzony algorytm nie ma możliwości zmiany kształtu okna przeszukiwań (umożliwia jedynie zmianę jego rozmiaru), które również powodowałyby spowolnienie jego działania. Wpływ rodzaju sekwencji na czas rozpoznania ilustruje Tabela 7. Na jej podstawie zauważyć można, że najgorszy wynik uzyskany został dla sekwencji 15 (Rysunek 106) przy bardzo dobrej skuteczności rozpoznania na poziomie 89 %. Spowolnienie działania algorytmu było wynikiem przysłonięciem obiektu oraz jego wyjściem poza zakres obejmowany przez urządzenie rejestrujące, które zwiększyło ilość wykonywanych operacji (zwiększenie okna przeszukiwań opisanego w rozdziale 5.5.1). Najlepszy czas lokalizacji obiektu został uzyskany dla sekwencji 7, lecz jednocześnie wystąpiła najniższa skuteczność rozpoznania. Algorytm szybko, lecz niewłaściwie rozpoznawał obiekt, co było spowodowane zakłóceniami występującymi na obrazach sekwencji.

Stworzony algorytm bardzo dobrze zaprezentował się w stosunku do badanych metod opisanych w rozdziale 4. Jego skuteczność była wyższa dla większości badanych sekwencji obrazów. W porównaniu do algorytmu Camshift gorsze wyniki uzyskane zostały jedynie dla sekwencji 9, co wynikało z lepszego dopasowania okna obrysowującego do kształtu obiektu szczególnie, gdy zbliżał się on do krawędzi obrazów sekwencji. Dla tej sekwencji również nieznacznie lepszy okazał się algorytm „Fast Template Matching", który oprócz tego uzyskał wyższą skuteczność podczas rozpoznania obiektu w sekwencji 10. Przysłonięcie piłki poprzez inny obiekt o stosunkowo bliskiej kolorystyce miało w tym wypadku znacznie mniejszy wpływ na właściwe rozpoznanie niż w wypadku stworzonego algorytmu. Metoda „Pyramid Lucas-Kanade Optical Flow" okazała się gorsza dla wszystkich badanych sekwencji.

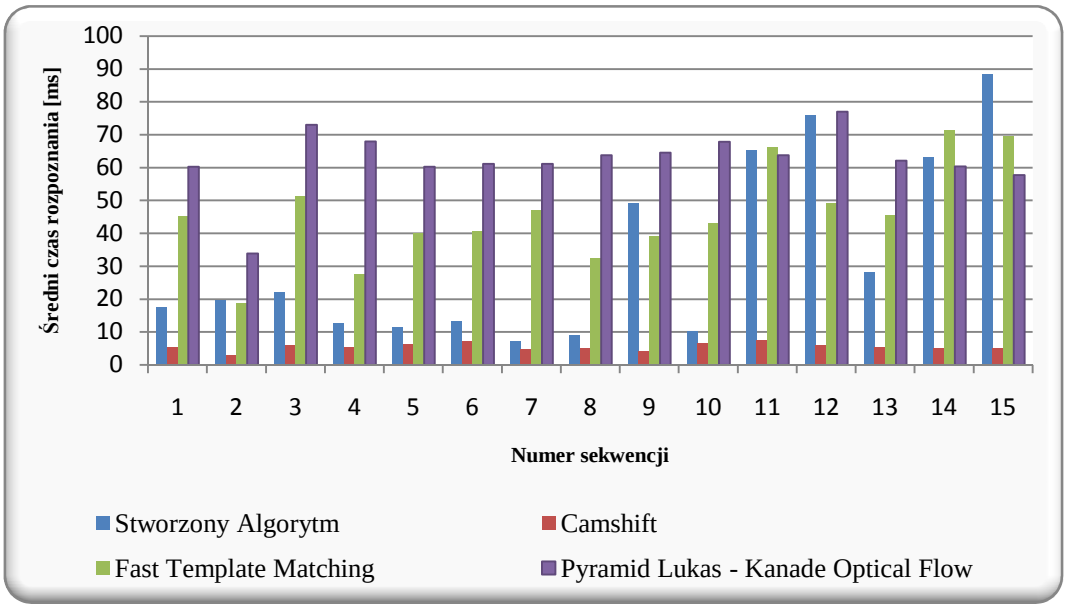
Poniżej przedstawione zostało porównanie skuteczności stworzonego algorytmu w stosunku do badanych algorytmów śledzenia obiektów. Dla lepszej ilustracji każde z nich umieszczone zostało na oddzielnym wykresie.



Rysunek 107. Porównanie skuteczności stworzonego algorytmu w stosunku do badanych algorytmów śledzenia obiektów

Pod względem czasu rozpoznania zdecydowanie najlepszy od stworzonego algorytmu okazał się Camshift. Uzyskał on lepsze wyniki dla wszystkich badanych sekwencji. W miarę wzrostu skomplikowania ruchu obiektu (to znaczy występowaniu przysłonięcia obiektu, jego rotacji itp.) pozostałe algorytmy miały zbliżone czasy lokalizacji. Trzeba jednak pamiętać, że mimo wzrostu czasu lokalizacji obiektu stworzony algorytm miał lepszą skuteczność rozpoznania w stosunku do innych algorytmów, a to w tym wypadku jest ważniejsze.

Poniżej przedstawione zostało porównanie średniego czasu rozpoznania dla wszystkich badanych algorytmów:



Rysunek 108. Porównanie średniego czasu rozpoznania stworzonego algorytmu w stosunku do badanych algorytmów śledzenia obiektów

6 Podsumowanie i wnioski

W części teoretycznej pracy opisano najbardziej popularne elementy przetwarzania obrazów znajdujących zastosowanie w śledzeniu obiektów w sekwencjach obrazów. Przedstawione zostały również powszechnie znane i stosowane metody lokalizacji obiektów.

Głównym celem praktycznym pracy było zbadanie dostępnych algorytmów śledzenia obiektów oraz stworzenie własnego algorytmu bazującego na podstawowych cechach. Do tego celu wykorzystany został język C++ z zastosowaniem biblioteki OpenCV.

Do badań wybranych algorytmów wykorzystane zostały funkcje dostępne w bibliotece OpenCV. Aby móc z nich w pełni skorzystać, należało odpowiednio przygotować badaną sekwencje oraz wydobyć z niej wymagane parametry używane podczas obliczeń. Każdy z algorytmów przetestowany został na kilkunastu sekwencjach pod względem czasu lokalizacji obiektu oraz skuteczności działania.

W kolejnej części pracy stworzony został algorytm śledzenia obiektów. W pierwszej jego wersji wybór obiektu odbywam się poprzez jego obrysowanie prostokątnym oknem, a następnie wyznaczeniem jego histogramu w przestrzeni RGB. Jego lokalizacja odbywała się poprzez porównanie histogramów fragmentów kolejnych klatek sekwencji z wzorcem i wyborze najlepszego wyniku porównania. Rozwiązanie to miało kilka podstawowych wad takich jak: występowanie dużych obszarów tła wokół obiektu zawartych we wzorze obiektu (błędny histogram), duży wpływ czynników zewnętrznych takich jak oświetlenie(zmiany wartości składowych RGB), czy również bardzo duża ilość operacji związana z przeszukiwaniem pełnych klatek sekwencji (negatywny wpływ na czas rozpoznania obiektu). W pierwszej kolejności zmieniony został sposób wyboru obiektu z oznaczenia prostokątnego na eliptyczne, które dawało większe możliwości uzyskania prawidłowego wzorca. W drugiej kolejności zmieniona została przestrzeń barw wykorzystywana do tworzenia histogramu z RGB na HSV, która ze względu na swoją charakterystykę ułatwia ograniczenie wpływu zmiennego oświetlenia. Aby w prosty sposób zmniejszyć czas

lokalizacji wprowadzone zostało ograniczenie okna przeszukiwań w zależności od wyniku porównania oraz zmienny skok pomiędzy kolejnymi iteracjami. Dodatkowo, aby zoptymalizować proces wyznaczenia maksymalnego dopasowania wzorca i fragmentu obrazu zastosowana została metoda Gaussa-Seida. Metoda ta mimo swojej prostej budowy wpłynęła pozytywnie na działanie algorytmu. Podczas tworzenia algorytmu sprawdzony został również jeden z najbardziej znanych predykatorów położenia, a mianowicie filtr Kalmana. Jego zastosowanie nie dało jednak dobrych efektów, gdyż obserwator nie nadążał nad zmianami kierunku ruchu obiektu i z tego powodu został pominięty w treści pracy.

Podsumowując stworzony algorytm sprawdził się dla większości badanych sekwencji, a jego skuteczność była wyższa niż innych badanych metod. Można go stosować dla różnych obiektów bez większych ograniczeń. Oczywiście w przyszłości można by rozszerzyć jego działanie wykorzystując większą ilość informacji o obiekcie, zastosować szybsze i dokładniejsze metody optymalizacji procesu lokalizacji, czy stosując różnego rodzaju predykcję jego położenia. Jednak stworzenie optymalnego rozwiązania dla wszystkich rodzajów obiektów i sekwencji jest bardzo trudne. Każda metoda sprawdza się w określonych sytuacjach i w tych przypadkach najczęściej jest wykorzystywana.

Literatura

[1] Bradski, G.; Kaehler, K.: Learning OpenCV Computer Vision with the OpenCV Library, O'Reilly Media Inc., 2008.

[2] Doros, M.; Przetwarzanie obrazów, WSISIZ, Warszawa, 2006/2007.

[3] Jahne, B.: Digital Image Processing, Springer – Verlag Berlin Heidelberg New York, 2002.

[4] Korzyńska, A.; Przytulska, M.: Przetwarzanie obrazów – ćwiczenia, Wydawnictwo PJWSTK, Warszawa, 2005.

[5] Iwanowski, M.; Skoneczny, S: Przetwarzanie i rozpoznawanie obrazów reskrypt, Program rozwojowy Politechniki Warszawskiej, 2010.

[6] Malina, W.; Smiatacz, M.: Metody Cyfrowego przetwarzania obrazów, Akademicka Oficyna Wydawnicza EXIT, Warszawa, 2005.

[7] Nieniewski, M: Morfologia matematyczna w przetwarzaniu obrazów Akademicka Oficyna Wydawnicza PLJ, Warszawa, 1998.

[8] Kukiełka G.; Woźnicki, J.: Praktyczne aspekty wykorzystania metod morfologii matematycznej w cyfrowym przetwarzaniu obrazów, Materiały konferencyjne z III sympozjum naukowego „Techniki przetwarzania obrazu", Serock, 1997.

[9] Buczyński, P.: Optymalna reprezentacja kolorów w analizie i przetwarzaniu obrazów komputerowych, Politechnika Warszawska, 2005.

[10] Watkins, C. D.; ,Sadun, A.; Marenka, S.: Nowoczesne Metody Przetwarzania Obrazów, Wydawnictwo Naukowo-Techniczne, Warszawa, 1995.

[11] Opis histogramu wielowymiarowego: [dostęp 15 września 2011] http://www.cs.put.poznan.pl/kkrawiec/zim-um/Przetwarzanie%20obrazow.pdf

[12] Tadeusiewicz, R.; Korohoda, P.: Komputerowa analiza i przetwarzanie obrazów, Wydawnictwo Fundacji Postępu Telekomunikacji, Kraków, 1997.

[13] Hu, W.; Tan, T.; Wang, L.; Maybank, S.: A survey on visual surveillance of object motion and behaviors, Systems, Man, and Cybernetics, Part C: Applications and Reviews, IEEE Transactions on Volume 34, Issue 3, s. 334–352, 2004.

[14] Smiatacz, M.: Automatyczna lokalizacja i śledzenie obiektów na obrazie, Politechnika Gdańska, 2005.

[15] Fermuller, C.; Shulman, D.; Aloimonos, Y.: The statistics of optical flow University of Maryland, 2000.

[16] Horn, B.K.P.; Schunck, B.G.: Determining optical flow, Artificial Intelligence 59, s. 81-87, 1993.

[17] Barron, J.L.; Fleet, D. J.; Beauchemin, S.S.: Performance of optical flow techniques, International Journal of Computer Vision, 12, 1, s. 43-77, 1994.

[18] Yilmaz, A.; Li, X.; Shah, M.: Object Contour Tracking Using Level Sets, University of Florida.

[19] Phang Chang, Phang Piau: Haar Wavelet Matrices Designation in Numerical Solution of Ordinary Differential Equations, IAENG International Journal of Applied Mathematics, s. 164-168, 2008.

[20] Viola, P.; Jones, M.J.: Rapid Object Detection Using a Boosted Cascade of Simple Features, IEEE CVPR, 2001.

[21] Shi, J.; Tomasi, C.: Good features to track, 9th IEEE Conference on Computer Vision and Pattern Recognition, 1994.

[22] Lowe, D.G.: Distinctive Image Features from Scale-Invariant Keypoints, International Journal of Computer Vision, 60, 2, s. 91-110, 2004.

[23] Lindeberg, T.: Scale-space theory: A basic tool for analysing structures at different scales Journal of Applied Statistics, 21,2, s. 224–270, 1994.

[24] Bay, H.; Tuytelaars, T.; Van Gool, L: SURF: Speeded up robust features, ECCV, 2006

[25] Ukrainitz, Y.; Sarel, B.: Meanshift Theory and Applications, Weizmann Institute of Science, 2004.

[26] Yilmaz, A.; Shafique, K.; Lobo, N.; Li, X.; Olson, T.; Shah, M.: Target Tracking in FLIR Imagery Using Mean Shift and Global Motion Compensation, IEEE Workshop on Computer Vision Beyond Visible Spectrum, 2001.

[27] Liu, Q.; Cai, C.; Ngan, K.N.; Li, H.: Camshift based real-time multiple faces match tracking International Symposium on Intelligent Signal Processing and Communication Systems, s. 221-224, 2007.

[28] Francois, A.: Camshift Tracker Design Experiments with Intel OpenCV and SAI, Institute for Robotics and Intelligent Systems University of Southern California, 2004.

[29] Yanjiang Wang; Baodi Liu; Wuli Wang: Human face tracking with adaptive facial orientation template by mean shift algorithm, Signal Processing, The 8th International Conference on Volume 3, s. 16-20, Beijing, 2006.

[30] Dawei Liang; Qingming Huang; Shuqiang Jiang; Hongxun Yao; Wen Gao: Mean-Shift Blob Tracking with Adaptive Feature Selection and Scale Adaptation, ICIP 2007. IEEE International Conference on Volume 3, s. 369 – 372, San Antonio, 2007.

[31] Fuh, C.; Maragos, P.: Region-based optical flow estimation, Proceedings CVPR'89, IEE Computer Society Conference, s. 130–135, San Diego, 1989.

[32] Amiaz, T.; Kiryati, N.: Dense discontinuous optical flow via contour-based segmentation, ICIP 2005. IEEE International Conference on Volume 3, s. 1264-1267, 2005.

[33] Chi-Cheng Cheng; Kun-Hsin Ho; Hui-Ting Li; Gwo-Long Lin: Image following using the feature-based optical flow approach, Proceedings of the 2002 IEEE International Symposium, s. 350–355, 2002.

[34] Jong-Ho Choi; Kang-Ho Lee; Kuk-Chan Cha; Jun-Sik Kwon; Dong-Wook Kim; Ho-Keun Song: Vehicle Tracking using Template Matching based on Feature Points, Information Reuse and Integration, 2006 IEEE International Conference, s. 573-577, Waikoloa Village, 2006

[35] Opis algorytmu Fast Template Matching [dostęp 20 maja 2010]: http://opencv.willowgarage.com/wiki/FastMatchTemplate

[36] Opis metody Gaussa – Seidla [dostęp 10 marca 2010]: http://www.kmg.ps.pl/opt/wyklad/bezgrad/1gauss.html