



A G H

AKADEMIA GÓRNICZO-HUTNICZA IM. STANISŁAWA STASZICA W KRAKOWIE

**WYDZIAŁ ELEKTROTECHNIKI, AUTOMATYKI,
INFORMATYKI I INŻYNIERII BIOMEDYCZNEJ**

Projekt dyplomowy

Sprzętowo-programowy system wizyjny wspomagający autonomiczne lądowanie drona.

Hardware-software vision system supporting the autonomous landing of a drone.

Autor: Jakub Kłosiński
Kierunek studiów: Automatyka i Robotyka
Opiekun pracy: dr inż. Tomasz Kryjak

Kraków, 2020

Serdecznie dziękuję . . .

Spis treści

1. Wprowadzenie	7
1.1. Cele pracy	7
1.2. Zawartość pracy.....	8
2. Metody przetwarzania obrazu i sposoby sterowania w procesie autonomicznego lądowania drona	9
3. Sprzęt wykorzystany w projekcie	13
3.1. Platforma obliczeniowa	13
3.2. Moduł rejestrujący obraz	14
3.3. Platforma statku powietrznego	14
3.4. Autopilot.....	16
4. Zaimplementowany system wspomagający autonomiczne lądowanie drona.....	17
4.1. Implementacja i ewaluacja modelu programowego	17
4.2. Implementacja sprzętowo-programowa.....	17
4.2.1. Integracja kamery i płytka	18
4.2.2. Konwersja z przestrzeni barw RGB do YCbCr	18
4.2.3. Binaryzacja.....	19
4.2.4. Mediana.....	19
4.2.5. Erozja i dylatacja.....	19
4.2.6. Prostokąt otaczający i pole powierzchni	20
4.2.7. Indeksacja.....	20
4.2.8. Integracja płytki z autopilotem.....	22
5. Testy i ocena wyników	25
5.1. Test wykrywania koloru znacznika w różnych przestrzeniach barw.....	25
5.1.1. Przestrzeń barw RGB	25
5.1.2. Przestrzeń barw YCbCr	25
5.1.3. Przestrzeń barw HSV	25
5.2. Wyznaczenie kąta widzenia kamery przy różnych ustawieniach rozdzielczości	25

5.3.	Test detekcji znacznika na obrazie	26
5.4.	Test regulacji położenia drona.....	29
5.5.	Lądownie na nieruchomym lądowisku	30
6.	Podsumowanie i kierunki dalszych prac.....	35

1. Wprowadzenie

Ostatnie lata pokazały wszechstronność i przydatność dronów w wielu dziedzinach przemysłu. Według raportu Skyward z 2018 roku, 1 na 10 przebadanych firm w Stanach Zjednoczonych używała bezzałogowych statków powietrznych. Aż 88 procent z nich odczuła pozytywne strony rozpoczęcia korzystania z nich w przeciągu roku lub krócej. Do najczęściej wymienianych zalet dronów należy zdobywanie większej ilości informacji, bardziej efektywna praca oraz oszczędzanie czasu. Udział dronów w rynku będzie się stale powiększał, gdyż 3 na 4 przedsiębiorców planuje zwiększać wydatki przeznaczane na operacje wykonywane przez drony [1]. O planach częstszego stosowania dronów może również świadczyć fakt wprowadzania regulacji prawnych - konieczności wyposażania dronów w nadajnik numeru identyfikacyjnego [2]. Odpowiednia regulacja zwiększy bezpieczeństwo lotów i może otworzyć drogę do masowego wykorzystania dronów.

Ważnym kierunkiem rozwoju bezzałogowych statków powietrznych jest ich autonomizacja, czyli przystosowanie do lotów bez nadzoru człowieka. Oznacza to wyposażenie dronów w systemy monitorowania otoczenia i implementację algorytmów sterowania działających w oparciu o zebrane informacje. Kluczową fazą autonomicznego lotu drona jest lądowanie. Bliskość ziemi wymaga dokładnego sterowania ruchem statku powietrznego. W przypadku, gdy dostarczane dane pochodzą z systemu wizyjnego, decydujące jest szybkie przetwarzanie obrazów. Podejście sekwencyjne, oparte o mikrokontrolery i procesory sygnałowe, często okazuje się nieskuteczne. Przekroczone zostają możliwości obliczeniowe takich układów, uniemożliwiając dodawanie kolejnych funkcjonalności związanych z autonomizacją. Układy FPGA (ang. *Field Programmable Gate Arrays*) są preferowaną platformą obliczeniową do realizacji zadań przetwarzania strumienia wideo. Oferują możliwość zrównoleglenia obliczeń oraz niewielkie opóźnienie przetwarzania.

1.1. Cele pracy

W ramach pracy należy stworzyć sprzętowo-programowy podsystem wizyjny, który będzie komponentem systemu autonomicznego lądowania drona. W pierwszym kroku należy przeprowadzić analizę literatury naukowej związanej z tematem - głównie dotyczącej detekcji i śledzenia oznaczonego lądowiska. W drugim etapie należy wykonać model programowy algorytmu (Matlab, Python, C++, OpenCV), który pozwala na wykrycie i śledzenie lądowiska. Dodatkowo należy wybrać sposób jego oznaczenia.

Zakłada się, że dron będzie wyposażony w kamerę o osi optycznej skierowanej prostopadle do podłożu, a lądowisko w dalszych etapach projektu będzie ruchome (umieszczone na pojeździe) - należy to uwzględnić przy rejestracji sekwencji testowych.

Ponadto należy sprawdzić, czy tylko na podstawie systemu wizyjnego możliwe jest określenie wysokości drona nad lądowiskiem - z dokładnością wystarczającą do przeprowadzenia procedury lądowania.

W trzecim etapie należy wspomniany system podzielić na część sprzętową i programową oraz zaimplementować w układzie Zynq SoC na wybranej karcie ewaluacyjnej. Wejściem powinien być obraz z kamery PCAM, a wyjściem sterowanie dla drona - położenie względem lądowiska oraz wysokość. Ostatnim etapem będzie próba integracji wykonanego podsystemu ze sterownikiem drona i wykonanie lądowania w sposób autonomiczny. W przypadku niesatysfakcjonującego pomiaru wysokości metodą wizyjną, możliwe jest użycie specjalistycznego czujnika laserowego. Kluczowym zagadnieniem będzie takie sterowanie dronem, aby lądowanie odbyło się bezpiecznie i w wyznaczonym miejscu, w szczególności w przypadku lądowiska zamontowanego na ruchomym pojeździe.

1.2. Zawartość pracy

2. Metody przetwarzania obrazu i sposoby sterowania w procesie autonomicznego lądowania drona

Wykonywanie różnorodnych scenariuszy misji przez drona wymaga osiągnięcia określonego poziomu autonomii. Bezzałogowy statek powietrzny musi być w stanie wystartować, nawigować oraz lądować bez bezpośredniego udziału człowieka. Przemieszczanie drona pomiędzy różnymi punktami nawigacyjnymi przebiega bezproblemowo, kiedy dostępny jest sygnał GPS. Kłopotu nie sprawia również start wykonywany bez pomocy operatora. Największe wyzwanie stanowi autonomiczne lądowanie drona. Platforma obliczeniowa zainstalowana na statku powietrznym musi być w stanie wykryć lądowisko oraz wysłać odpowiednie sygnały sterujące. Efektem działania algorytmu sterującego musi być zajęcie dogodnej pozycji do obniżenia lotu i ostatecznie przyziemienie w określonym punkcie.

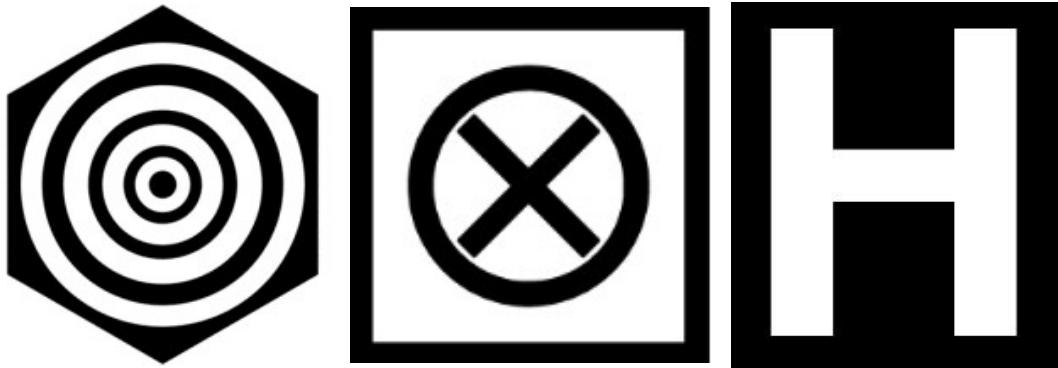
Zadania autonomicznego lądowania drona można podzielić na dwie kategorie:

- lądowanie na platformie pozostającej w bezruchu względem ziemi,
- lądowanie na poruszającej się platformie.

W przypadku lądowania na oznaczonym stacjonarnym lądowisku, głównym problemem jest skuteczna detekcja znacznika. Do realizacji tego zadania niezbędne jest wykonanie szeregu kroków. Przykładowe rozwiązanie z pracy [3] obejmuje:

- zamianę obrazu kolorowego na obraz w skali szarości,
- binaryzację ze stałym progiem,
- indeksację,
- odrzucenie obiektów o liczbie pikseli mniejszej niż zadany próg,
- identyfikację znacznika.

Wartości progów binaryzacji oraz odrzucenia małych obiektów zostały dobrane eksperymentalnie. Aby dodatkowo wyeliminować biały szum oraz zakłócenia typu „sól i pieprz” w pracy [4] wprowadzono operację mediany z oknem 3x3 na obrazie w skali szarości. Dodatkowo wprowadzono ograniczenie na maksymalny rozmiar obiektu. W algorytmie przedstawionym w pracy [5] binaryzacji dokonano w przestrzeni barw HSV (ang. *Hue Saturation Value*), na podstawie składowych S i V. Pozwoliło to na przeprowadzenie segmentacji niezależnej od koloru. Przed rozpoznawaniem kształtów, na obrazie binarnym



(a) Znacznik użyty w pracy [3]

(b) Znacznik użyty w pracy [6]

(c) Znacznik użyty w pracy [7]

Rys. 2.1. Porównanie znaczników użytych w pracach [3],[6] i [7]

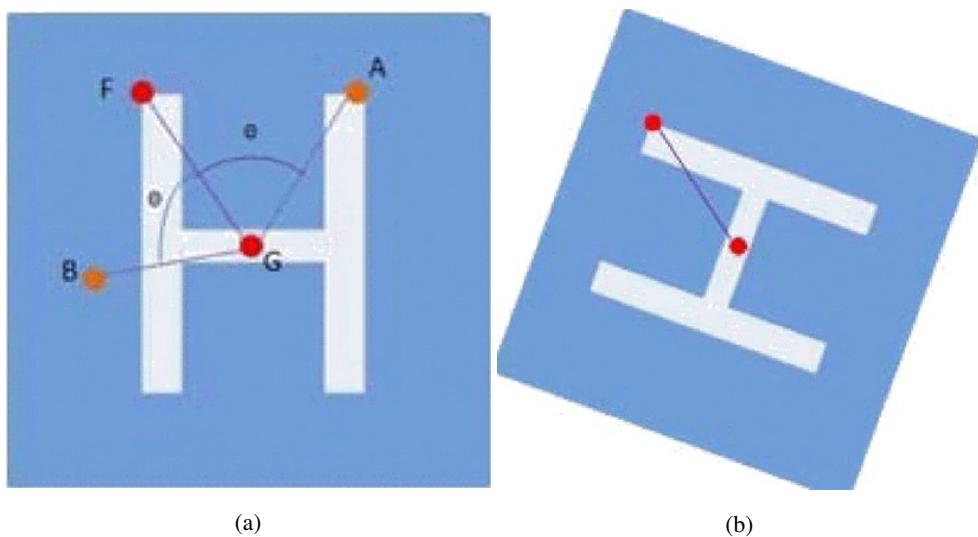
dokonano kolejno erozji, detekcji krawędzi oraz dylatacji. Wykonanie erozji pozwoliło na eliminację z obrazu małych grup pikseli.

Istotnym elementem planowania autonomicznego lądowania drona jest wybór znacznika lądowiska. W pracy [6] przedstawiono marker złożony z trzech figur: kwadratu, koła i krzyża. Znacznik pokazano na Rys. 2.1b. Zaimplementowany został algorytm detekcji lądowiska polegający na kolejnym wykrywaniu wymienionych figur, co pozwoliło na coraz lepsze określenie położenia celu.

Prostsze rozwiązanie zostało pokazane w pracy[7], gdzie znacznik ma kształt litery H (rysunek 2.1c) Odległość drona od lądowiska obliczana jest na podstawie liczby pikseli dzielących środek ciężkości znacznika od środka obrazu. Opisano tam również metodę wyznaczania orientacji drona względem takiego markera. Polega ona na znalezieniu piksela najbardziej odległego od środka ciężkości, a następnie wykreśleniu linii przechodzącej przez te dwa punkty. Otrzymana linia nie określa jednak pozycji znacznika w sposób jednoznaczny (Rys. 2.2). Do określenia, czy zachodzi przypadek przedstawiony na Rys. 2.2a, czy też 2.2b, obliczono kąt θ pomiędzy dwoma narożnymi punktami markera. Kąt obliczany jest na podstawie zapisanego obrazu znacznika. Następnie wykorzystując znalezioną wcześniej linię, obliczane są współrzędne tych dwóch punktów (punkty A i B na Rys. 2.2a). Kolejną czynnością jest sprawdzenie, który z punktów leży bliżej obszaru markera. Jeśli jest to punkt A - zachodzi przypadek z Rys. 2.2a, jeśli B - Rys. 2.2b.

Inny znacznik został użyty w pracy [3]. Składa się on z czterech pierścieni na czarnym tle (Rys. 2.1a) Każdy pierścień posiada unikalny stosunek promienia wewnętrznego do zewnętrznego i stanowi oddzielny obiekt dla systemu wizyjnego. Obiekty, które nie mają dokładnie jednej dziury wewnętrznej, są pomijane. Każdy z pozostałych jest ostatecznie identyfikowany za pomocą współczynnika kształtu. Zaletą takiego znacznika jest możliwość dołożenia kolejnych, większych pierścieni, jeśli lądowisko ma być widoczne z większej wysokości.

W procesie autonomicznego lądowania drona, na podstawie wyznaczonej odległości od celu, do UAV wysyłane są sygnały sterujące. W pracy [8] wykorzystano trzy regulatory PID, do kontroli prędkości drona. Dzięki nim równocześnie minimalizowana jest każda z trzech składowych wektora położenia.



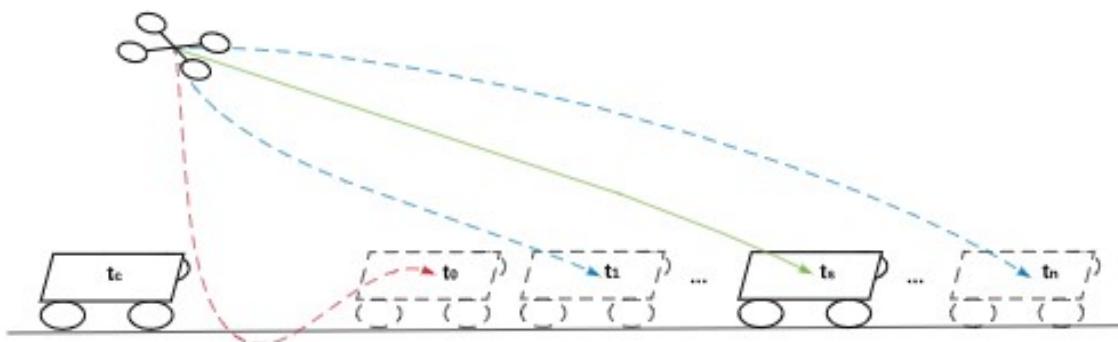
Rys. 2.2. Ilustracja określania pozycji znacznika w pracy [7]

W pracy [3] zwrócono uwagę na błędy spowodowane pochyleniem i przekątne drona. Przed wysłaniem sygnału do regulatora wykonywana jest korekta uchybu na podstawie pomiaru kąta pochylenia i przekątne. Krok ten był potrzebny z uwagi na sztywne umieszczenie kamery na ramie drona.

W przypadku wykonywania lądowania na poruszającej się platformie, do opisanego wcześniej problemu detekcji znacznika dochodzi również konieczność implementacji bardziej skomplikowanego algorytmu sterowania. Wysyłane sygnały sterujące muszą uwzględniać ruch lądowiska. Przemieszczanie drona powinno nadążać za zmianą położenia platformy. W takim przypadku układ regulacji jest układem śledzącym.

W [6] użyto algorytmu pozwalającego na predykcję zachowania celu. Wykorzystano model dynamiczny celu oraz rozszerzony filtr Kalmana. Na Rys. 2.3 przedstawiono schemat wybierania trajektorii. System planuje kilka możliwych dróg dotarcia do lądowiska. Każda z nich ma początek w aktualnej pozycji drona, a kończy się w przewidzianym położeniu platformy. Przyszły stan poruszającego się celu jest określany na podstawie jego modelu dynamicznego, począwszy od ostatniej dostępnej estymaty z filtru Kalmana. Trajektorie wymagające sterowania spoza dostępnego zakresu lub kolidujące z przeszkodami, są odrzucone (trajektoria oznaczona czerwoną przerywaną linią). Spośród wszystkich możliwych dróg (niebieskie przerywane linie) wybierana jest ta wymagająca użycia najmniejszej ilości energii. Zadanie optymalizacji jest wykonywane przy wykorzystaniu szybkiej wielomianowej generacji trajektorii minimalizującej trzecią pochodną położenia.

Podsumowując, wykonanie lądowania na statycznym lądowisku wymaga skutecznej detekcji znacznika znajdującego się na platformie, który określa jego lokalizację. Znalezienie względnej pozycji markera umożliwia przekazanie informacji o uchybie do regulatora sterującego dronem. Do urządzenia wykonawczego, czyli silników, dociera ostatecznie informacja o pożądanej wartości ciągu, która spowoduje zbliżenie do celu. Po ustabilizowaniu unoszenia drona nad lądowiskiem następuje polecenie stopniowego



Rys. 2.3. Przykład wybierania trajektorii z pracy [6].

obniżania lotu, aż do zetknięcia z ziemią. W przypadku lądowania na ruchomym celu nie ma możliwości doprowadzenia do zawisu nad platformą. Należy przeprowadzić predykcję zachowania lądowiska, wygenerować trajektorie i użyć jednej z metod optymalizacji do wyboru najlepszej z nich.

3. Sprzęt wykorzystany w projekcie

3.1. Platforma obliczeniowa

W pracy wykorzystano platformę Digilent Zybo Z7-20 z układem Zynq SoC (ang. *System on Chip*) XC7Z020-1CLG400C. Układ dostępny na karcie określa się jako heterogeniczny tj. stanowi połączenie części rekonfigurowalnej (PL – ang. *programmable logic*) oraz systemu procesorowego z dwurdzeniowym procesorem ARM Cortex-A9 taktowanym z częstotliwością 667 MHz (PS – ang. *processing system*).

Na Rys. 3.1 przedstawiono architekturę układu Zynq SoC. System procesorowy zaznaczony został na zielono, natomiast część rekonfigurowalna na żółto. Część PS zawiera wiele komponentów, między innymi:

- wydajne kontrolery 1Gb Ethernet, USB 2.0, SDIO (ang. Secure Digital Input Output),
- kontrolery SPI, UART, CAN, I2C,
- kontroler pamięci DDR3,
- magistralę *Advanced Microcontroller Bus Architecture Interconnect* (AMBA),
- inne kontrolery peryferiów z wejściami/wyjściami multipleksowanymi do 54 dedykowanych pinów MIO (ang. Multiplexed Input Output),
- piny EMIO (ang. Extended MIO) pozwalające na podłączenie komponentów poprzez część PL.

Kontrolery peryferiów są podłączone do części PS poprzez magistralę AMBA w trybie *slave*. W ten sposób uzyskują dostęp do rejestrów odczytu/zapisu, adresowalnych w pamięci procesora. Również część rekonfigurowalna jest połączona z magistralą AMBA poprzez porty AXI jako *slave*. Daje to możliwość szybkiej komunikacji między układem FPGA, a procesorem. Ponadto, moduły zaimplementowane w części PL mogą wywoływać przerwania w procesorze i otrzymywać dostęp DMA (ang. *Direct Memory Access*) do pamięci DDR3.

Poniżej przedstawiono komponenty części rekonfigurowalnej:

- 53 200 tablic LUT (ang. *Look-up Table*),
- 106 400 przerzutników *flip-flop*,

- 630 KB pamięci blokowej RAM,
- 6 obszarów zarządzania zegarami CMT (ang. *Clock Management Tiles*),
- konwerter analogowo-cyfrowy.

Do pozostałych części układu ZYBO Z7-20 należą między innymi:

- łącznik kamery PCAM ze wsparciem MIPI CSI-2
- wejściowy oraz wyjściowy port HDMI
- slot na kartę SD
- 6 portów PMOD
- 4 przełączniki
- 5 diod LED

3.2. Moduł rejestrujący obraz

W projekcie wykorzystano kolorową kamerę Digilent PCAM 5C. Jest to moduł przeznaczony do użycia z dedykowanymi płytami rozwojowymi. Jednym z kompatybilnych układów jest ZYBO Z7-20. Właściwości modułu przedstawiono poniżej:

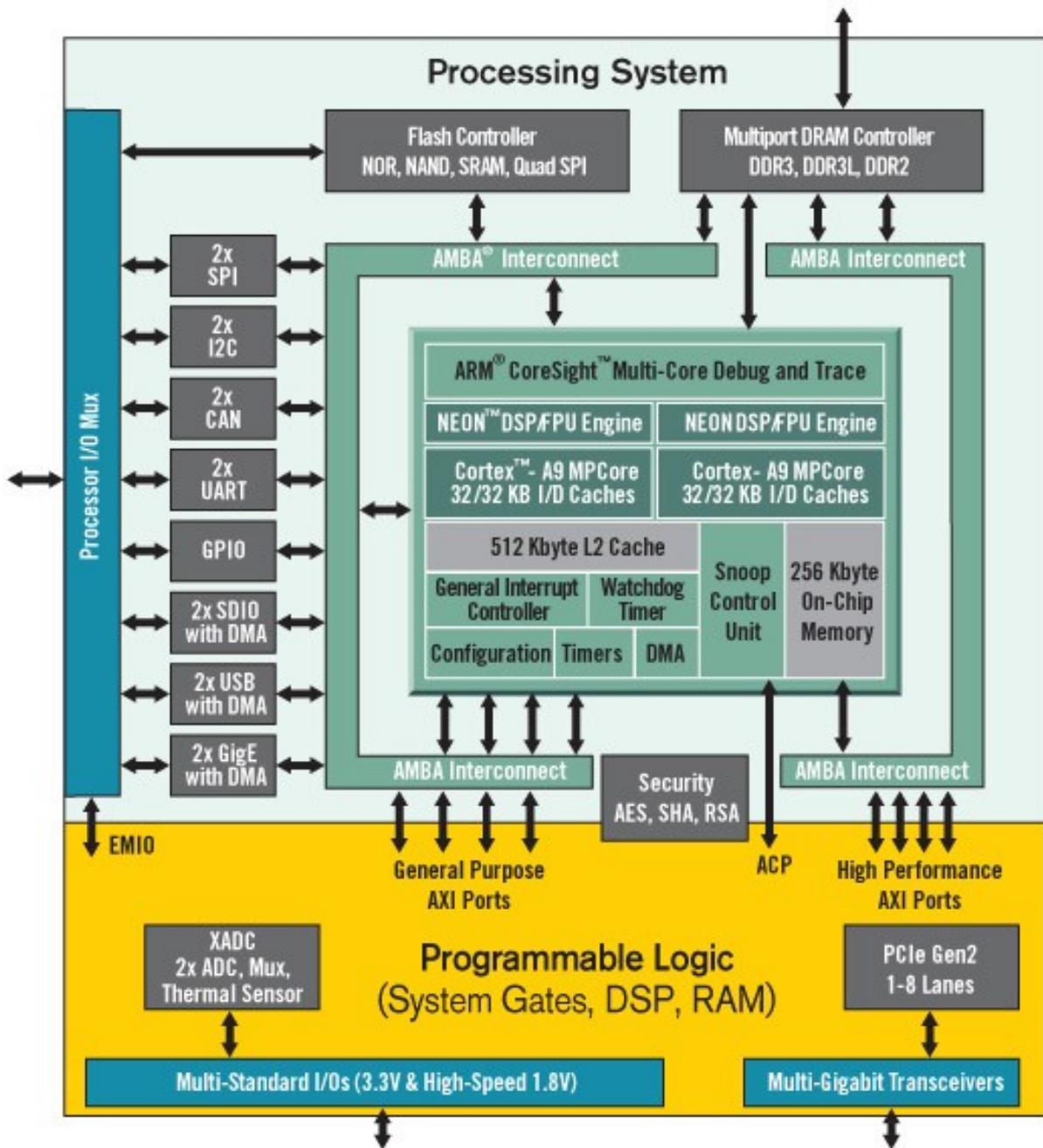
- Rozdzielcość: 5 MPx,
- Matryca: OV5640,
- Interfejs danych: MIPI CSI-2
- Złącze: 15-pinowe FFC

Sposób podłączenia kamery i układu ZYBO Z7-20 przedstawione zostały w sekcji 4.2.1.

3.3. Platforma statku powietrznego

Zdecydowano się na dron sześciowirnikowy, znajdujący się na wyposażeniu studenckiego koła naukowego AVADER. Zbudowanie drona było częścią innego projektu [10]. Elementy użytego bezzałogowego statku powietrznego to:

- rama DJI F550,
- śmigła o średnicy równej 22,86 cm oraz skoku 12,7 cm,



Rys. 3.1. Architektura układu Zynq SoC [9].

- silniki DJI 2312/960KV z kontrolerami 420 LITE,
- czterokomorowa bateria LiPo o nominalnym napięciu 14,8 V i pojemności 6450mAh,
- nadajnik radiowy FrSky Taranis X9D Plus, odbiornik FrSky X8D,
- sterownik 3DR Pixhawk.

3.4. Autopilot

Urządzeniem bezpośrednio komunikującym się z kontrolerami silników drona jest autopilot. W pracy wykorzystano sterownik Pixhawk, będący popularnym kontrolerem lotu ogólnego przeznaczenia dostępnym na otwartej licencji. Jego główne parametry to:

- procesor Cortex-M4F z zegarem 168 MHz,
- czujniki: akcelerometr, żyroskop, kompas magnetyczny, barometr i zewnętrzny moduł GPS,
- interfejsy: UART, CAN, I2C, SPI,
- wejście karty SD,
- zewnętrzny przełącznik bezpieczeństwa uzbrajania,
- wielokolorowa dioda pokazująca stan pracy

Konfiguracja sterownika jest możliwa przy użyciu programu Mission Planner. Jest to aplikacja przeznaczona dla stacji naziemnej umożliwiająca:

- strojenie czujników autopilota,
- monitorowanie stanu sterownika (uzbrojenie silników, orientacja statku powietrznego),
- planowanie, zapisywanie i wgrywanie planów autonomicznych misji statku powietrznego,
- pobieranie i analizowanie dzienników misji.

Sterownik umożliwia pracę w różnych trybach. W zależności od aktywnego tybu, zadania wykonywane przez autopilota nieco się różnią. Z punktu widzenia autonomicznego lotu, najbardziej interesującym trybem jest tryb GUIDED. Umożliwia on bowiem wydawanie dronowi poleceń. Zazwyczaj komunikacja przebiega drogą radiową ze stacją naziemną (przy użyciu programu Mission Planner), lecz wysyłanie komend przez urządzenie umieszczone na platformie drona również jest możliwe.

Kwestię połączenia autopilota i układu ZYBO Z7-20 opisano w sekcji 4.2.8.

4. Zaimplementowany system wspomagający autonomiczne lądowanie drona

W rozdziale omówiono zagadnienia związane ze szczegółami implementacji systemu. Ważną rolę w tworzeniu kolejnych modułów odegrał model programowy, czyli program napisany w dowolnym języku programowania i wykonywany na komputerze PC, którego działanie oddaje funkcje modułu twozonego w sprzęcie. Porównanie wyników działania modelu programowego i symulacji modułu daje informację o poprawności implementacji.

Następnie opisano wszystkie części przetwarzania sygnału, od przesłania obrazu przez kamerę, do wysyłania żądanych komend do autopilota.

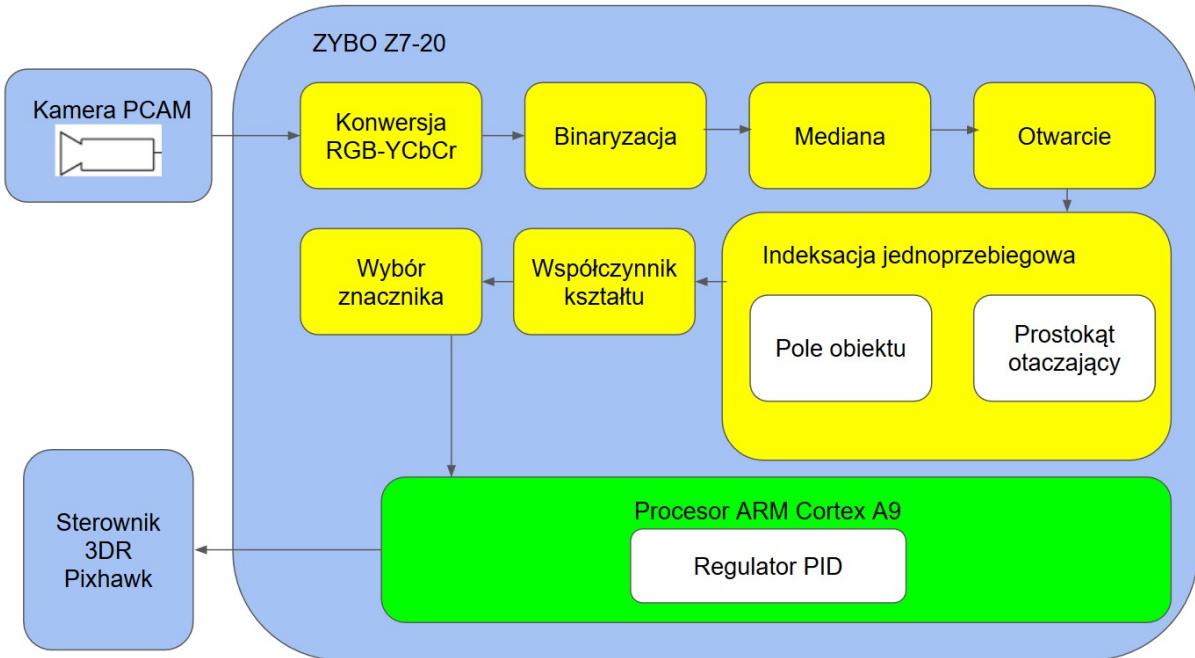
4.1. Implementacja i ewaluacja modelu programowego

Model programowy został napisany w pakiecie Matlab przy wykorzystaniu funkcji dostępnych w bibliotece *Image Processing Toolbox*. Pozwoliło to na szybkie prototypowanie systemu wizyjnego.

Aby możliwe było wprowadzanie ramek obrazu do modelu, należało umożliwić akwizycję obrazów na kartę SD. Uaktyniono interfejs procesora SD0, do projektu w SDK dodano bibliotekę „xilffs” i, korzystając z funkcji systemu plików opisanych w [11], napisano zapis ramek do pliku na karcie SD. Ze względu na prostotę pliku, składającego się jedynie z nagłówka oraz wartości kolejnych składowych RGB, zdecydowano się na format ppm.

4.2. Implementacja sprzętowo-programowa

Na Rys. 4.1 przedstawiono ogólny schemat implementacji systemu. Na niebiesko zaznaczone zostały komponenty sprzętowe, na żółto oznaczono moduły zaimplementowane w części rekonfigurowalnej, natomiast na zielono zaznaczono system procesorowy. Z kamery trafia do układu sygnał wizyjny, który następnie jest przetwarzany przez poszczególne moduły. Po wyborze znacznika informacja o jego pozycji i uchybie regulacji trafia do procesora. System procesorowy wysyła komendy zmiany sterujące do autopilota. Zadaniem układu regulacji jest przesunięcie drona nad lądowisko, umożliwiając wykonanie lądowania. W dalszej części rozdziału omówiono poszczególne komponenty systemu.



Rys. 4.1. Schemat przedstawiający zaimplementowany system.

4.2.1. Integracja kamery i płytka

Producent na swojej stronie internetowej zapewnia projekt demonstracyjny połączenia kamery i płytka [12]. Przez port szeregowy możliwa jest zmiana rozdzielczości, szybkości akwizycji ramek, współczynnika korekcji gamma, ustawień balansu bieli. Możliwe są następujące opcje dotyczące dwóch pierwszych parametrów:

- 1280 x 720, 60 fps,
- 1920 x 1080, 15 fps,
- 1920 x 1080, 30 fps.

Testy pokazały również, że dla rozdzielczości 1280 x 720 większy jest kąt widzenia kamery. Kwestię tę opisano w sekcji ???. Ze względu na powyższy fakt oraz przyspieszenie obliczeń, zdecydowano się na najmniejszą dostępną rozdzielcość.

Przeprowadzanie syntezy i implementacji projektu możliwe było przy użyciu darmowego oprogramowania Vivado oraz SDK w wersji WebPack – używano wersji 2018.2.

Pobrany projekt stanowił bazę do dalszych prac.

4.2.2. Konwersja z przestrzeni barw RGB do YCbCr

Piksel w przestrzeni barw YCbCr opisują trzy składowe: Y (luminancja), Cb (chrominancja, która wyraża różnicę między luminancją, a kolorem niebieskim) oraz Cr (chrominancja, która wyraża różnicę

między luminancją, a kolorem czerwonym). Zaletą stosowania tej przestrzeni barw jest oddzielenie sygnału luminancji od sygnałów chrominancji, pozwalające na dokonywanie przetwarzania sygnału przy mniejszej zależności od oświetlenia. Konwersję z przestrzeni barw RGB do YCbCr wykonano zgodnie ze wzorem 4.1. Przy implementacji wykorzystano sprzętowe mnożarki oraz sumatory.

$$\begin{bmatrix} Y \\ Cb \\ Cr \end{bmatrix} = \begin{bmatrix} 0,299 & 0,587 & 0,114 \\ -0,168736 & -0,331264 & 0,5 \\ 0,5 & -0,418688 & 0,081312 \end{bmatrix} \begin{bmatrix} R \\ G \\ B \end{bmatrix} + \begin{bmatrix} 0 \\ 128 \\ 128 \end{bmatrix} \quad (4.1)$$

4.2.3. Binaryzacja

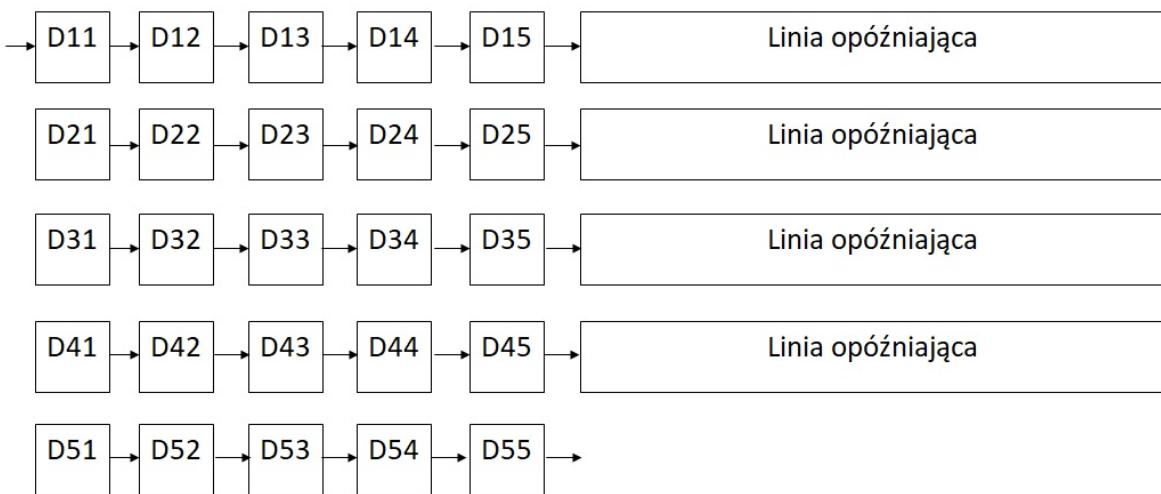
Piksel wyjściowy otrzymywał wartość maksymalną (kolor biały), jeśli wartości Cb i Cr mieściły się pomiędzy wyznaczonymi eksperymentalnie progami. W innym przypadku pikselowi przypisywana była wartość 0 (kolor czarny). Do szybkiej zmiany progów wykorzystano komunikację PS-PL z wykorzystaniem rejestrów AXI - progi mogły być zmieniane z poziomu terminala i testy nie wymagały ponownych implementacji.

4.2.4. Mediana

Mediana to operacja kontekstowa, w której pikselowi wyjściowemu przypisuje się wartość średniewariancjalną uporządkowanego zbioru wartości pikseli z otoczenia piksela wejściowego. W przypadku działania na obrazie binarnym operacja taka może być przeprowadzona przez obliczenie sumy wartości pikseli wewnętrz kontekstu, a następnie porównanie jej z połową maksymalnej wartości tej sumy. Ze względu na łatwość implementacji oraz zadowalające wyniki filtracji, zdecydowano się na rozpatrywanie kontekstu w kształcie kwadratu o boku 5 pikseli. Spowodowało to konieczność zapamiętywania kontekstu piksela w 25 rejestrach oraz 4 linii obrazu w długich liniach opóźniających zbudowanych w oparciu o pamięć BRAM. Schemat wyznaczania kontekstu przedstawiono na rysunku 4.2. Sygnały synchronizacji zostały doklejone do wartości piksela i w przedstawionej strukturze przesuwają się razem z nim. Sumę wyliczano w 2 etapach, dodając najpierw elementy w wierszach, potem sumując wyniki.

4.2.5. Erozja i dylatacja

Erozja i dylatacja to operacje morfologiczne, w których pikselowi wyjściowemu przypisuje się wartość odpowiednio najmniejszego i największego piksela w sąsiedztwie piksela wejściowego. Sąsiedztwo piksela określa kształt i rozmiar elementu strukturalnego. Podobnie jak w przypadku mediany, zdecydowano się na kwadrat o boku 5 pikseli. Moduł erozji ustawia wartość piksela wyjściowego na maksymalną wartość, gdy w sąsiedztwie znajdują się same białe piksele. W module dylatacji zwracana jest wartość maksymalna, gdy przynajmniej jeden piksel w sąsiedztwie ma kolor biały.



Rys. 4.2. Schemat wyznaczania kontekstu dla mediany, erozji i dylatacji.

4.2.6. Prostokąt otaczający i pole powierzchni

Znalezienie prostokąta otaczającego sprowadza się do wyznaczenia skrajnych jego punktów na górze, dole, po lewej oraz prawej stronie. Na podstawie sygnałów synchronizacji oraz wymiarów obrazka wyznaczano współrzędne aktualnie przetwarzanego piksela. Jeśli jest to piksel należący do obiektu, następuje inkrementacja jego pola powierzchni. Do odpowiednich rejestrów trafiają wówczas również wartości współrzędnych piksela, jeśli wykraczają poza aktualną zawartość rejestrów:

- do rejestru zawierającego górnego bok prostokąta trafi współrzędna wierszowa, jeśli będzie ona mniejsza od aktualnej,
- do rejestru zawierającego dolny bok prostokąta trafi współrzędna wierszowa, jeśli będzie ona większa od aktualnej,
- do rejestru zawierającego lewy bok prostokąta trafi współrzędna kolumnowa, jeśli będzie ona mniejsza od aktualnej,
- do rejestru zawierającego prawy bok prostokąta trafi współrzędna kolumnowa, jeśli będzie ona większa od aktualnej,

Obliczeń pola powierzchni i współrzędnych prostokąta otaczającego nie wykonywano w oddzielnym module, lecz stanowiły one część modułu indeksacji.

4.2.7. Indeksacja

Indeksacja to operacja pozwalająca na wyodrębnienie z obrazu poszczególnych obiektów. Obiekty rozumiane są jako grupy połączonych ze sobą pikseli. Zazwyczaj na wejście modułu indeksacji podawany jest obraz zbinaryzowany, natomiast na wyjściu pojawia się obraz, na którym wartość pikseli odpowiada przypisanej do danego obiektu etykiecie. Rozważane jest otoczenie każdego piksela, składające

się z trzech pikseli nad nim oraz jednego po lewej stronie, tak jak zostało to przedstawione na rysunku 4.3. Indeksacji dokonuje się bezpośrednio na obrazie wejściowym. Podczas iteracji po wszystkich pikselach, w przypadku znalezienia piksela należącego do któregoś z obiektów, może zajść jeden z trzech przypadków:

- (a) w otoczeniu piksela znajdują się tylko piksele należące do tła,
- (b) otoczenie zawiera jeden lub więcej pikseli, którym została wcześniej przypisana taka sama etykieta L ,
- (c) w otoczeniu znajdują się piksele posiadające różne etykiety.

D5	D4	D3
D2	D1	

Rys. 4.3. Sąsiedztwo piksela brane pod uwagę przy indeksacji

W pierwszym przypadku pikselowi zostaje przypisana nowa etykieta. Gdy spełniony jest warunek b , punkt otrzymuje etykietę L , natomiast jeśli zachodzi przypadek c , przypisywana jest mniejsza z etykiet. W ten sposób otrzymuje się obraz wstępnie poetykietowany. Najczęściej posiada on więcej przypisanych etykiet, niż obiektów. Dlatego istnieje koniecznośćłączenia ze sobą pewnych etykiet przy użyciu tablicy sklejeń. Tablica ta zawiera informację, które etykiety powinny zostać złączone. W przypadku a do tablicy sklejeń na pozycji odpowiadającej etykiecie zapisywana jest etykieta, natomiast gdy zachodzi opcja c etykietę mniejszą zapisuje się pod indeksem większym. Do sklejenia etykiet potrzebna jest druga iteracja, tym razem po obrazie wstępnie poetykietowanym.

Powyższy fakt jest główną przeszkodą w łatwym wykonaniu takiego algorytmu w systemie potokowym. Bez zapamiętywania całej ramki, w przypadku sklejania etykiet, niemożliwy jest powrót do wcześniej przetwarzanych pikseli. Pomimo tego, możliwe jest obliczenie pewnych cech obiektów, takich jak: pole, współrzędne środka ciężkości, prostokąt otaczający. W pracy [13] podano sposób, w jaki można tego dokonać. Opiera się on na scaleniu nie samych wartości pikseli, ale obliczanych na bieżąco parametrów obiektu.

Implementacja w języku Verilog rodzi dodatkowo trudności związane z określeniem przypadku istnienia tej samej lub różnych etykiet w otoczeniu piksela. O ile wykrycie przypadku a jest łatwe, to przypadki b i c wymagały rozważenia kilku możliwości. Zdecydowano się na wykrywanie ich za pomocą flag bitowych. Na ich podstawie wnioskowano o zachodzącym aktualnie przypadku oraz wskazywano, od którego piksela z otoczenia powinna zostać przepisana etykieta. Zgodnie z opisanymi wcześniej

zasadami uzupełniano również tablicę sklejeń. Oprócz tego, na bieżąco obliczano prostokąt otaczający oraz liczbę pikseli należących do każdego ze znalezionych obiektów.

Po poetykietowaniu całej ramki obrazu wykorzystano tablicę sklejeń do złączenia obliczanych na bieżąco cech. Ten etap algorytmu zaimplementowano jako maszynę stanów:

- Stan 0 – Oczekiwanie na sygnał końca ramki wyznaczany na podstawie synchronizacji pionowej. W momencie wykrycia sygnału następuje rejestrowanie tablicy sklejeń i obliczonych parametrów (w następnym taktie zostaną one zresetowane), zerowanie tablic wypełnianych w kolejnym etapie oraz przejście do stanu 1.
- Stan 1 – Iteracja po tablicy sklejeń i uzupełnianie rzeczywistych wartości cech obiektów.
- Stan 2 – Uporządkowanie tablic z wyznaczonymi parametrami.
- Stan 3 – Obliczenie pola prostokąta otaczającego dla każdego znalezionego obiektu. Wykorzystywana jest mnożarka o latencji 3.
- Stan 4 – Szukanie obiektu spełniającego warunki minimalnej wielkości pola powierzchni oraz stosunku pola prostokąta otaczającego do pola obiektu. W przypadku znalezienia takiego kształtu, na wyjście trafiają współrzędne jego środka ciężkości. Jeśli żądany kształt nie zostanie znaleziony, informacja o tym również pojawi się na wyjściu.

Schemat przetwarzania danych po każdej ramce pokazano na rysunku 4.4.

Główną trudnością w implementacji opisanego algorytmu w układzie FPGA jest stosunkowo duże zapotrzebowanie na zasoby sprzętowe. Należy zarezerwować miejsce na cechy każdego potencjalnego obiektu. Ogranicza to liczbę możliwych etykiet. Zarezerwowanie miejsca dla 30 obiektów nie przekroczyło możliwości układu ZYBO. Badania pokazały, że jest to wystarczająca liczba etykiet do sprawnego działania toru wizyjnego i ostatecznie zaimplementowano moduł w tej wersji.

4.2.8. Integracja płytki z autopilotem

Tablica sklejeń:

1	1
2	1
3	1
4	4
5	5
6	5
7	4

Znalezione położenia lewego boku prostokąta:

1	15
2	10
3	20
4	25
5	22
6	17
7	8

Najmniejsze wartości dla każdego obiektu:

1	10
2	0
3	0
4	8
5	17
6	0
7	0

Wartości przygotowane do wyboru znalezionejego obiektu:

1	10
2	8
3	17
4	0
5	0
6	0
7	0

Rys. 4.4. Schemat procesu przetwarzania informacji po każdej ramce obrazu z przykładowymi danymi. Tablica sklejeń daje informację o konieczności sklejenia etykiet 1, 2, 3. Ponieważ przykład dotyczy lewego boku prostokąta, spośród liczb 15, 10, 20 (odpowiadającym etykietom 1, 2, 3) wybierana jest najmniejsza. Ostatni krok przetwarzania to usunięcie środkowych zer z wektora.

5. Testy i ocena wyników

5.1. Test wykrywania koloru znacznika w różnych przestrzeniach barw

W celu zbadania możliwości wykrycia koloru znacznika przygotowano 3 fotografie wykonane kamerą Digilent PCAM 5C. Każde ze zdjęć przedstawia czerwone koło (kształt znacznika był przy tym trochę mniej istotny) i zostało zrobione przy innym oświetleniu. Zdjęcia zostały przedstawione na rysunku 5.1. Sprawdzono trzy systemy barw: RGB, YCbCr i HSV. Badania przeprowadzono wykorzystując normalizowane histogramy oraz dostępne w programie Matlab narzędzie Data Cursor.

5.1.1. Przestrzeń barw RGB

Histogramy składowych RGB dla różnych poziomów oświetlenia przedstawiono na rysunku 5.2. Analiza histogramów prowadzi do wniosku, że wartości składowych silnie zależą od oświetlenia i detekcja koloru w tej przestrzeni barw jest bardzo trudna.

5.1.2. Przestrzeń barw YCbCr

5.1.3. Przestrzeń barw HSV

5.2. Wyznaczenie kąta widzenia kamery przy różnych ustawieniach rozdzielczości

Do zbadania kąta widzenia kamery wykorzystano moduł wyznaczający prostopadłe linie przechodzące przez środek obrazu. Następnie umieszczono kamerę na wysokości 49 cm i skierowano ją w dół. Odczytanie odległości od środka obrazu do jego krawędzi w poziomie i pionie pozwoliło na wyznaczenie kątów widzenia kamery. Skorzystano ze wzoru (5.1).

$$\alpha = \arctan \frac{l}{h} \quad (5.1)$$

gdzie:

α – kąt widzenia kamery,

h – wysokość, na jakiej umieszczona jest kamera,



Rys. 5.1. Zdjęcia, które wykorzystano do przeprowadzenia testu wykrywania koloru w różnych systemach barw

Tabela 5.1. Wartości kąta widzenia kamery w zależności od rozdzielczości

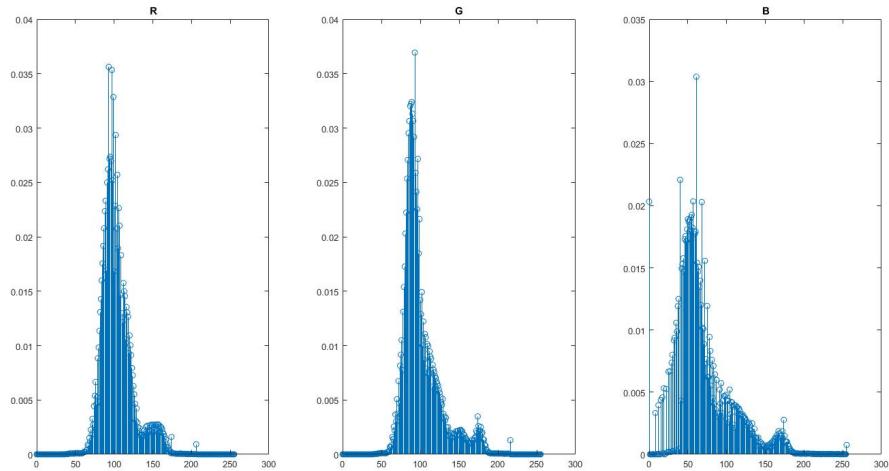
Rozdzielczość	Kąt widzenia kamery w pionie w stopniach	Kąt widzenia kamery w poziomie w stopniach
1920 x 1080	14	27
1280 x 720	19	38

l – odległość środka obrazu od jego krawędzi.

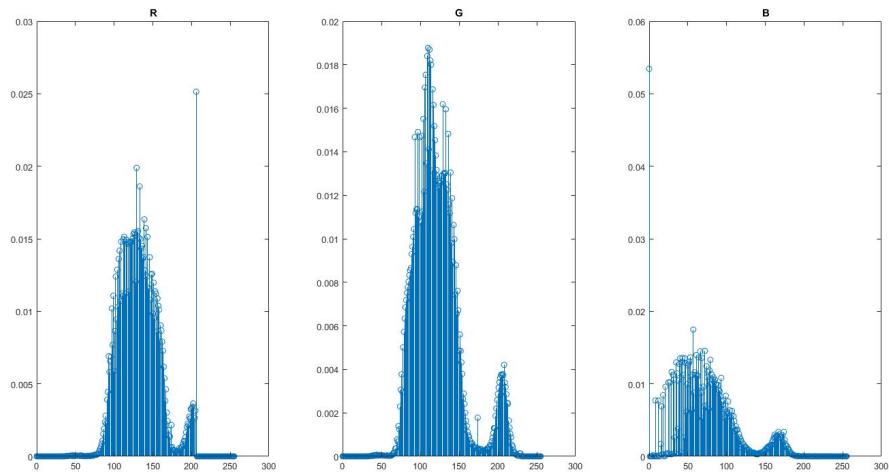
Na podstawie obrazu przedstawionego na rysunku 5.9 obliczono kąty widzenia kamery w pionie i poziomie dla rozdzielczości 1920 x 1080. Dla rozdzielczości 1280 x 720 skorzystano z obrazu pokazanego na rysunku 5.10. Korzystając ze wzoru (5.1) otrzymano przybliżone wyniki, przedstawione w tabeli 5.1.

5.3. Test detekcji znacznika na obrazie

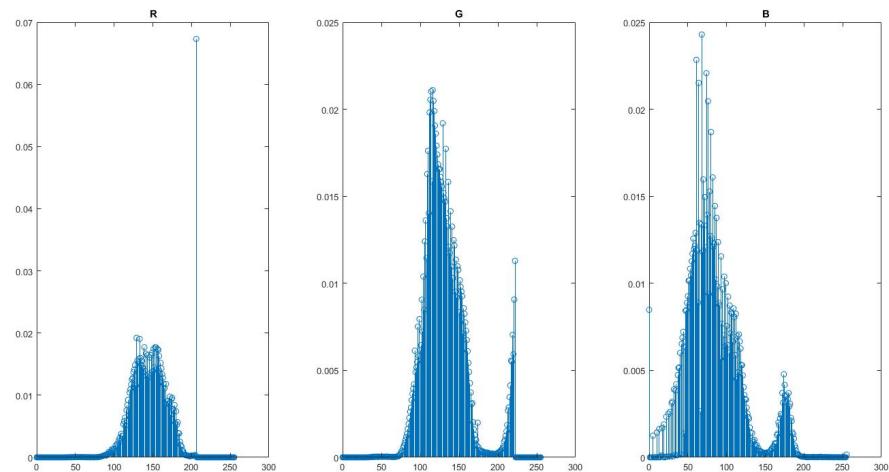
Na rysunku 5.11 przedstawiono zdjęcie znacznika wykonane z wysokości jednego metra. Zastosowanie binaryzacji ze stałym progiem (rysunek 5.12) pozwoliło na oddzielenie znacznika od tła. Na obrazie znajdują się jednak inne niewielkie obszary (zakłócenia), które zostały wykryte. Wykorzystanie mediany spowodowało znaczne zmniejszenie tych obszarów, jednakże na rysunku 5.13 widoczne są nadal małe grupy białych pikseli nienależących do markera. Zastosowanie morfologicznego otwarcia,



(a) Histogram obrazu o najmniejszym z badanych oświetleń

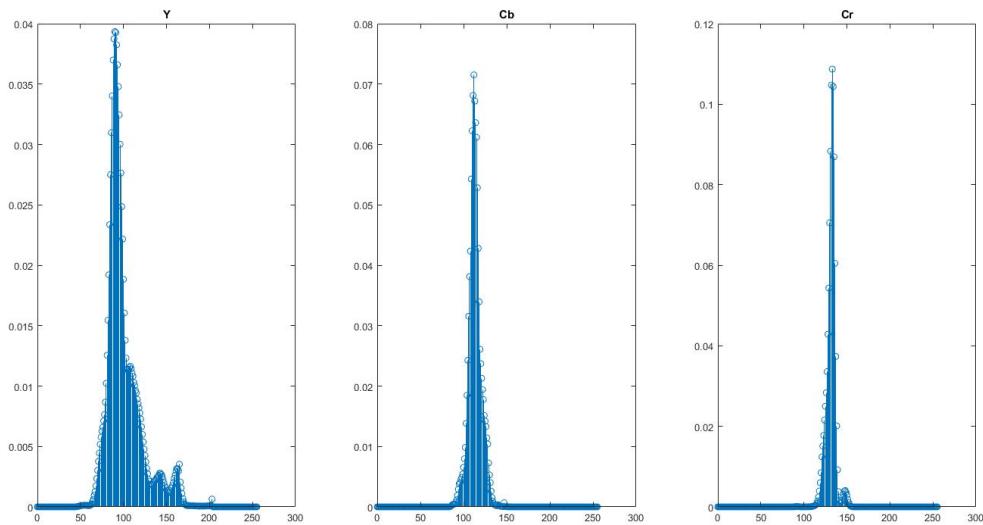


(b) Histogram obrazu o średnim oświetleniu

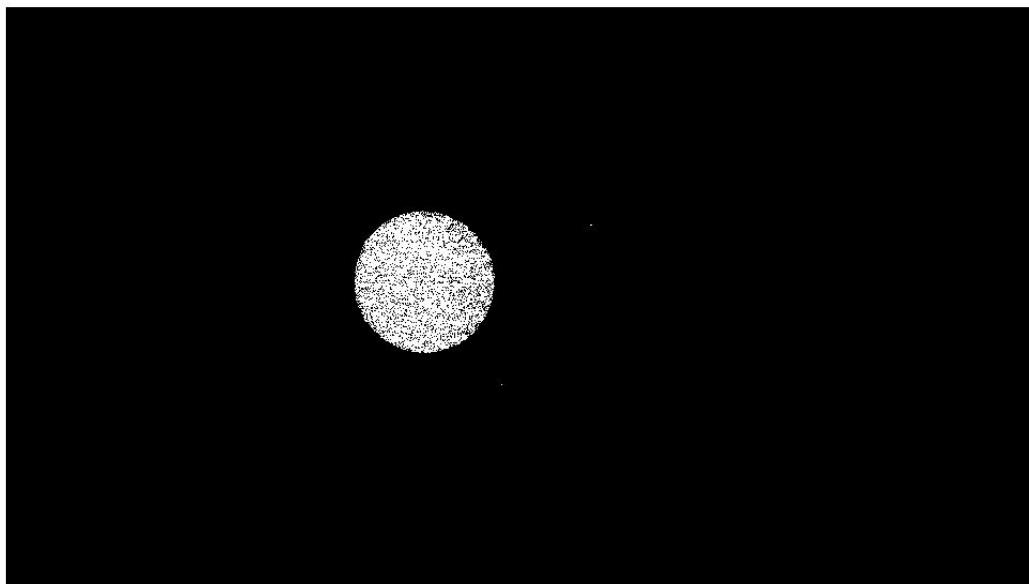


(c) Histogram obrazu o największym z badanych oświetleń

Rys. 5.2. Histogramy obrazów w przestrzeni RGB

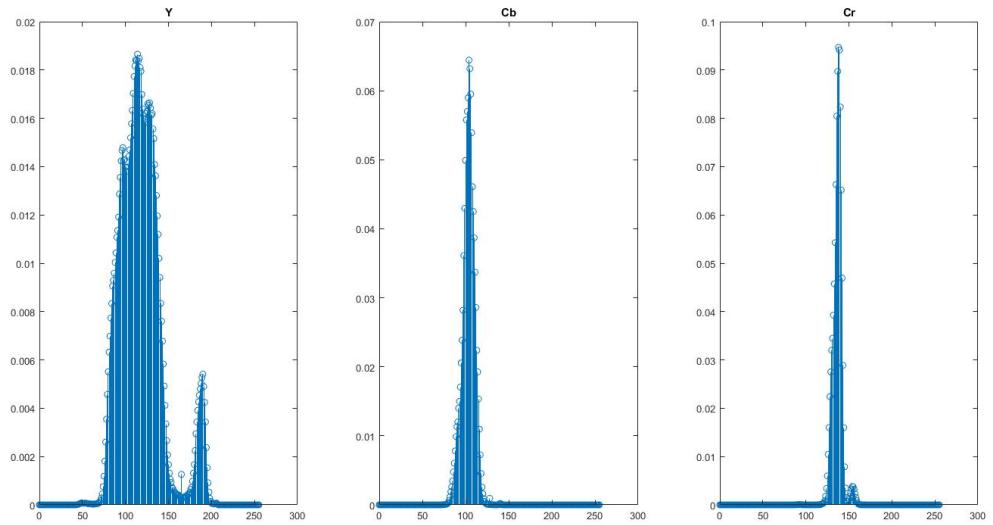


Rys. 5.3. Histogram obrazu wykonanego przy najmniejszym z badanych oświetleń

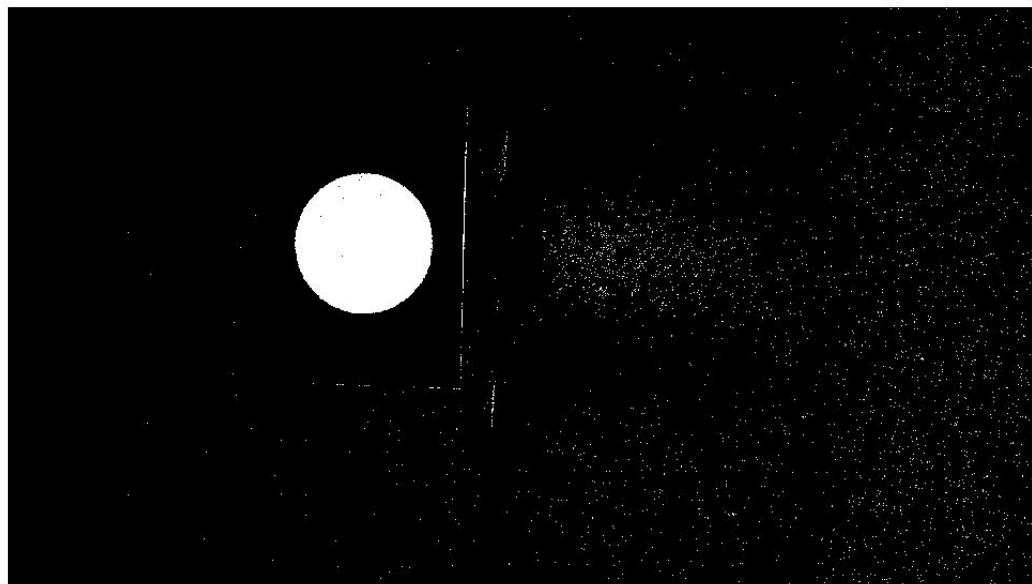


Rys. 5.4. Binaryzacja obrazu wykonanego przy najmniejszym z badanych oświetleń

składającego się z erozji i dylatacji, pozwoliło na całkowite wyeliminowanie błędnych białych pikseli. Na koniec wykorzystano również operację zamknięcia, polegającą na wykonaniu najpierw dylatacji, a potem erozji. Na rysunkach 5.14 i 5.15, przedstawiających rezultaty kolejno otwarcia i zamknięcia, nie widać większych różnic. W pewnych sytuacjach jednak otwarcie prowadzi do podziału obiektu na kilka części. W takich przypadkach wykonanie zamknięcia może ponownie połączyć obiekty.



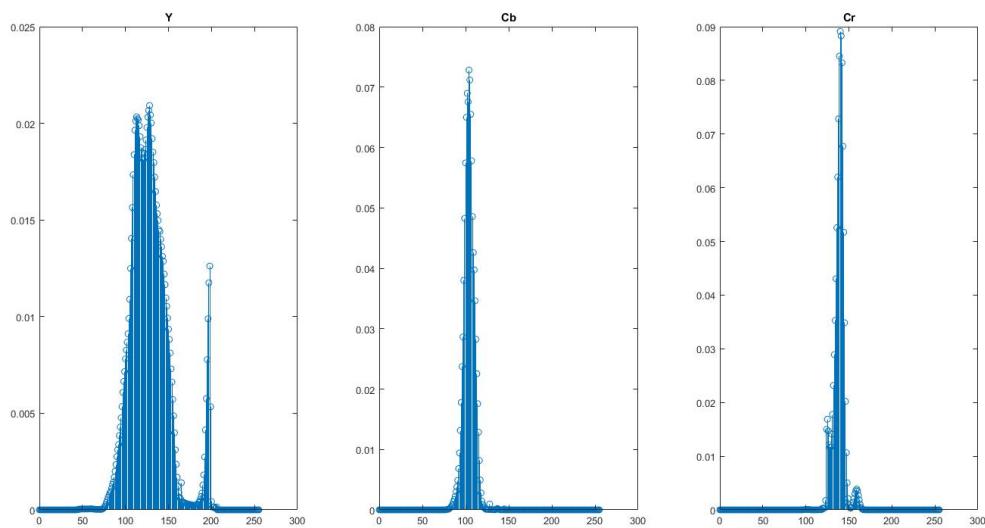
Rys. 5.5. Histogram obrazu wykonanego przy średnim oświetleniu



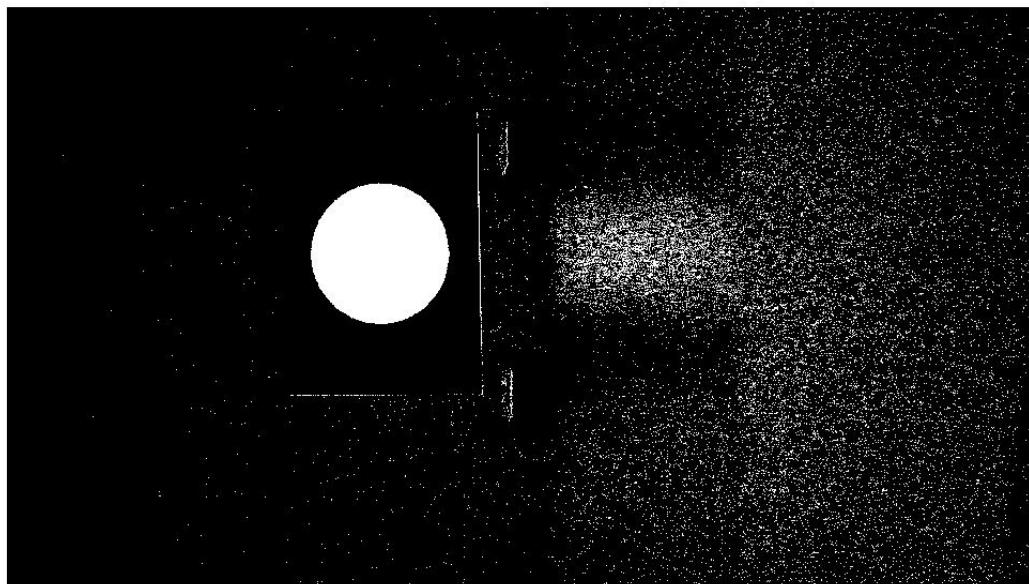
Rys. 5.6. Binaryzacja obrazu wykonanego przy średnim oświetleniu

5.4. Test regulacji położenia drona

— Test będzie polegał na zadawaniu prędkości w x i y proporcjonalnej do liczby pikseli dzielających środek obrazu od aktualnego położenia znacznika. Celem będzie doprowadzenie do sytuacji stabilnego unoszenia drona nad znacznikiem. Rejestrowanie uchybu w x i y i ich zapis na kartę SD pozwoli na sporządzenie wykresów tych wartości w czasie.



Rys. 5.7. Histogram obrazu wykonanego przy największym z badanych oświetleń



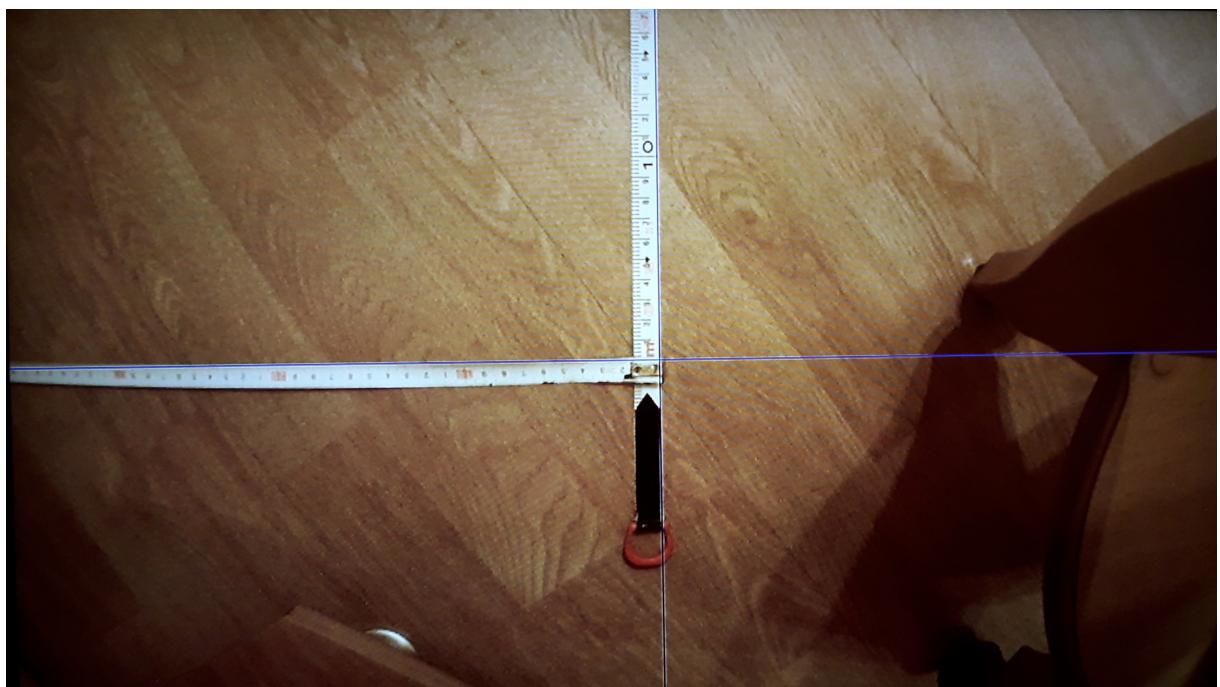
Rys. 5.8. Binaryzacja obrazu wykonanego przy największym z badanych oświetleń

5.5. Lądowanie na nieruchomym lądowisku

— Do poprzedniego testu zostanie dodana funkcjonalność lądowania.



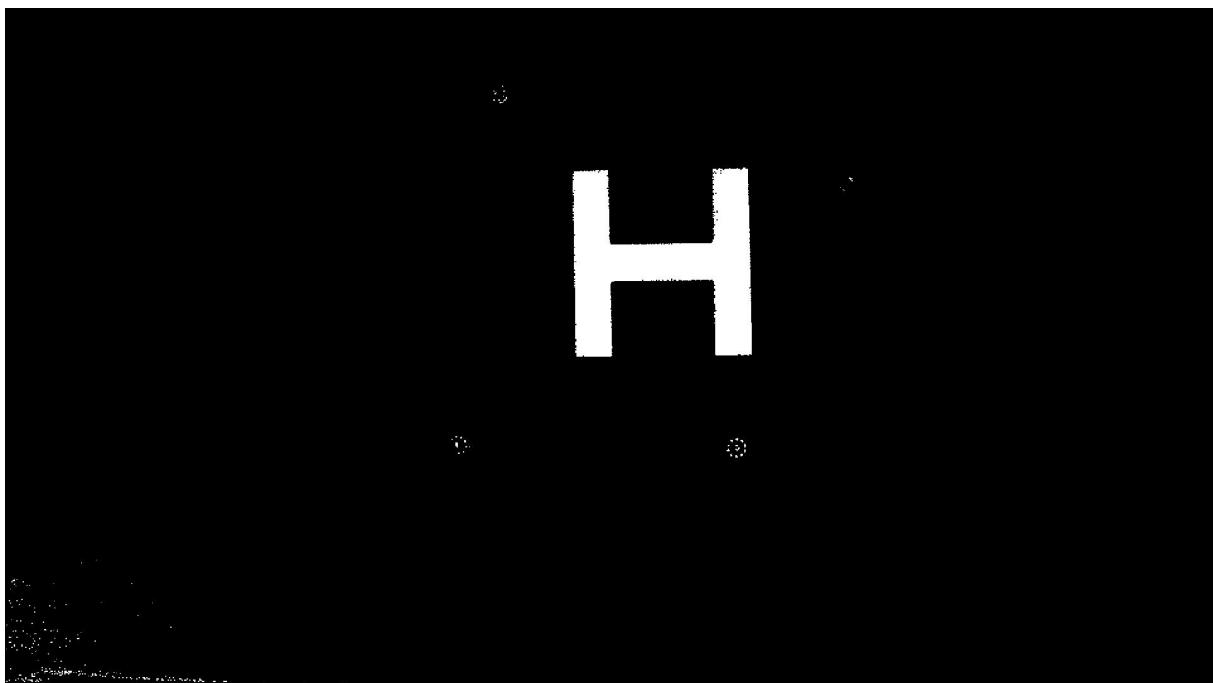
Rys. 5.9. Obraz służący do wyznaczenia kąta widzenia kamery dla rozdzielcości 1920 x 1080.



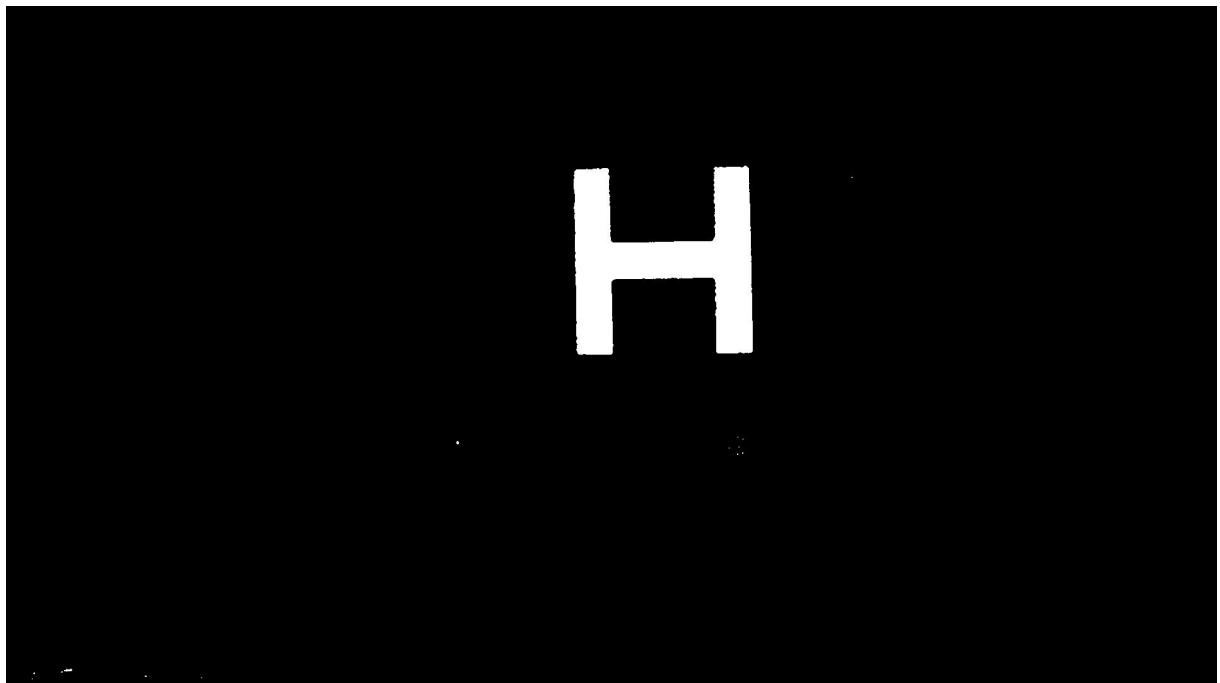
Rys. 5.10. Obraz służący do wyznaczenia kąta widzenia kamery dla rozdzielcości 1280 x 720.



Rys. 5.11. Obraz oryginalny



Rys. 5.12. Obraz po binaryzacji



Rys. 5.13. Obraz po medianie



Rys. 5.14. Obraz po otwarciu



Rys. 5.15. Obraz po zamknięciu

6. Podsumowanie i kierunki dalszych prac

Bibliografia

- [1] <http://go.skyward.io/rs/902-SIU-382/images/2018StateofDrones.pdf>. Dostęp: 2020-01-18.
- [2] <https://spectrum.ieee.org/aerospace/aviation/us-commercial-drone-deliveries-will-finally-be-a-thing-in-2020>. Dostęp: 2020-01-18.
- [3] S. Lange, N. Sunderhauf i P. Protzel. „A vision based onboard approach for landing and position control of an autonomous multirotor UAV in GPS-denied environments”. W: *2009 International Conference on Advanced Robotics*. 2009, s. 1–6.
- [4] Y. Fan, S. Haiqing i W. Hong. „A Vision-Based Algorithm for Landing Unmanned Aerial Vehicles”. W: *2008 International Conference on Computer Science and Software Engineering*. T. 1. 2008, s. 993–996. DOI: [10.1109/CSSE.2008.309](https://doi.org/10.1109/CSSE.2008.309).
- [5] A. Price i inni. *Real time object detection for an unmanned aerial vehicle using an FPGA based vision system*. Hubei, China: IEEE, 2008.
- [6] Davide Falanga i inni. *Vision-based Autonomous Quadrotor Landing on a Moving Platform*. Shanghai, China: IEEE, 2017.
- [7] Yang Fan, Shi Haiqing i Wang Hong. „A Vision-Based Algorithm for Landing Unmanned Aerial Vehicles”. W: *2008 International Conference on Computer Science and Software Engineering*. INSPEC Accession Number: 10426910. Hubei, China: IEEE, 2008, s. 995.
- [8] V. Sudevan, A. Shukla i H. Karki. „Vision based autonomous landing of an Unmanned Aerial Vehicle on a stationary target”. W: *2017 17th International Conference on Control, Automation and Systems (ICCAS)*. 2017, s. 362–367. DOI: [10.23919/ICCAS.2017.8204466](https://doi.org/10.23919/ICCAS.2017.8204466).
- [9] <https://reference.digilentinc.com/reference/programmable-logic/zybo-z7/reference-manual>. Dostęp: 2020-01-18.
- [10] M. Mach. „Wbudowany system wizyjny do śledzenia obiektów dla potrzeb nawigacji bezzałogowego statku powietrznego (UAV)”. Praca magisterska. AGH, 2017.
- [11] http://elm-chan.org/fsw/ff/00index_e.html. Dostęp: 2019-11-03.
- [12] <https://reference.digilentinc.com/learn/programmable-logic/tutorials/zybo-z7-pcam-5c-demo/start>. Dostęp: 2019-11-03.

- [13] Abdul Malik i in. „Real-time Component Labelling with Centre of Gravity Calculation on FPGA”. W: (sty. 2011).