AKADEMIA GÓRNICZO-HUTNICZA

im. St. Staszica w Krakowie

WEAIiE, Katedra Automatyki

Laboratorium Biocybernetyki

Przedmiot:	**Wieloprocesorowe Systemy Wizyjne.**			
			PR15wsw511	
Temat projektu:				
	Wizyjny pomiar obiektów szybko zmiennych			
Wykonali:				Michał Ciszewski Łukasz Dudek
			Automatyka i Robotyka, V rok, II stopień	
Rok akademicki		2015/2016, sem. zimowy		
Prowadzący		Dr inż. Mirosław Jabłoński		

wersja 1.0

Kraków, styczeń, 2016

Spis treści

1. Abstrakt

Niniejszy raport jest sprawozdaniem z prac prowadzonych przez dwuosobowy zespół nad wizyjnymi pomiarami szybko zmiennych obiektów. Głównym celem projektu było stworzenie stanowiska badawczego, gdzie zadaniem układu wyzwalającego byłoby automatyczne wykrywanie ruchu obrotowego odpowiedniego obiektu (np. wentylatora z komputera klasy PC) i aktywacja kamery. Ramki z kamery byłyby przechwytywane przez oprogramowanie na komputerze klasy PC, które wyznaczałoby prędkość obrotu obiektu. Część sprzętowa miała być oparta na układzie FPGA bądź SoC, który byłby w stanie mierzyć opóźnienia występujące w całym torze pomiarowym. Oprogramowanie miało być napisane z wykorzystaniem akceleracji obliczeń biblioteką OpenCL.

W trakcie prac nad projektem udało się zrealizować automatyczne wykrywanie ruchu wentylatora oraz wysyłanie sygnału wyzwalającego do kamery. Powstało również oprogramowanie, które jest w stanie badać obecność znacznika na obracającym się obiekcie i, znając częstotliwość akwizycji, wyznaczyć na tej podstawie prędkość obrotową obiektu. Nie udało się zrealizować części związanej z pomiarem opóźnień ani połączenia części sprzętowej – układu wyzwalającego – z częścią programową – oprogramowaniem do analizy obrazów – nie działa ono automatycznie, tzn. nie potrafi połączyć się z kamerą i zebrać z niej kolejnych ramek obrazu.

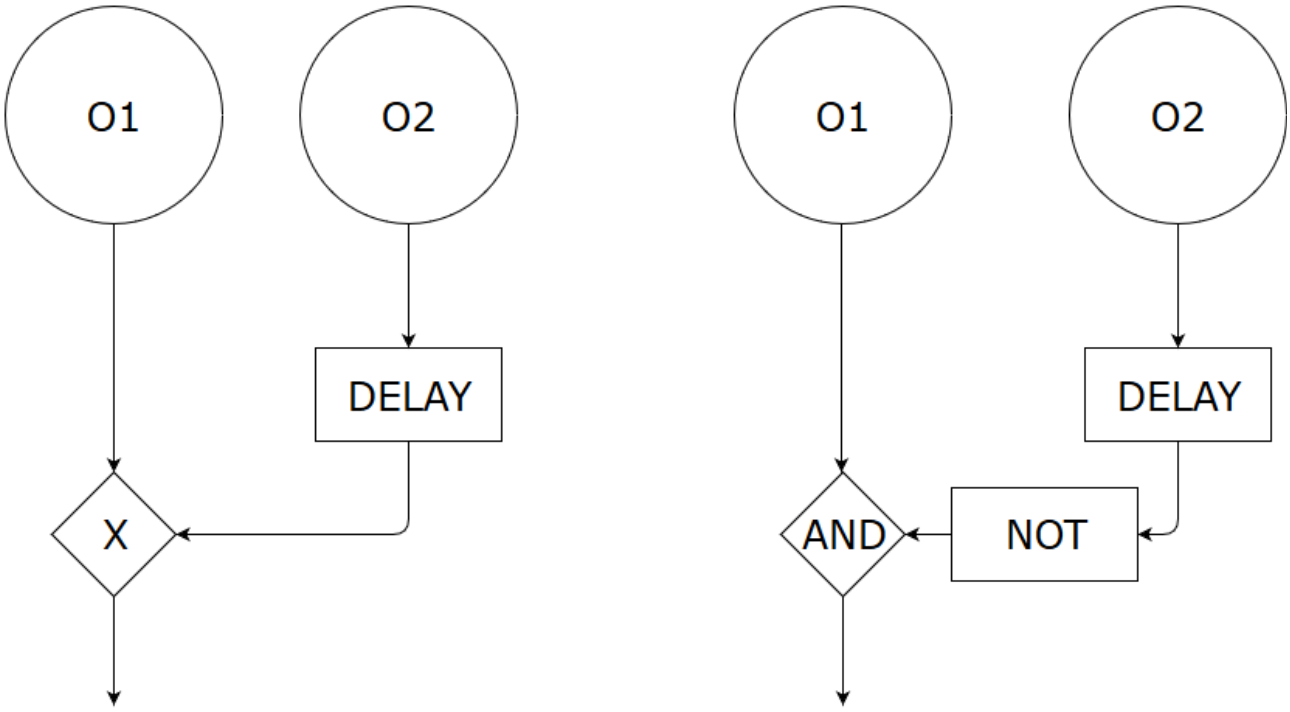
2. Wstęp

Postawiony w temacie projektu problem jest zagadnieniem głównie praktycznym. Jego rozwiązanie ma dostarczyć stanowiska pomiarowego do badania prędkości obrotowej obiektów. Ma się ono składać z kamery przemysłowej, układu wyzwalającego, komputera klasy PC oraz samego badanego obiektu. Potrzebne będzie również źródło oświetlenia i mocowania dla kamery i obiektu. Działanie układu powinno być następujące: układ wyzwalający wykrywa początek ruchu obiektu i wysyła sygnał wyzwolenia akwizycji do kamery. Kamera wysyła zebrane ramki do komputera klasy PC, gdzie odpowiednie oprogramowanie je analizuje i na tej podstawie wylicza prędkość obrotową obiektu. Układ wyzwalający powinien również móc zaobserwować opóźnienia w działaniu układu, w szczególności:

- opóźnienie między rozpoczęciem ruchu a wysłaniem sygnału wyzwalającego do kamery,
- opóźnienie między wysłaniem sygnału wyzwalającego a otrzymaniem od kamery potwierdzenia rozpoczęcia akwizycji,
- opcjonalnie również opóźnienia między początkiem akwizycji a ustaleniem prędkości obrotowej przez oprogramowanie na komputerze.

Wszystkie te opóźnienia mogą być również mierzone względem pierwszego zdarzenia, a więc rozpoczęcia ruchu.

Zakładamy, że oś obrotu obiektu jest równoległa do kierunku ustawienia kamery, prędkość obrotowa może być zmienna. Dodatkowo, i obiekt, i kamera są w stanie spoczynku (nie poruszają się względem siebie). Część programowa powinna być w stanie przetwarzać kolejne ramki obrazu z wystarczającą prędkością, aby wyznaczyć dokładnie prędkość obiektu oraz nie „zgubić" żadnej z nich. Część sprzętowa (układ wyzwalający) powinien móc prowadzić detekcję ruchu oraz zdarzeń czasowych opisanych powyżej z zaniedbywalnie małymi opóźnieniami.



Rysunek 1: Schematy opóźnionego porównania. Źródło: [1].

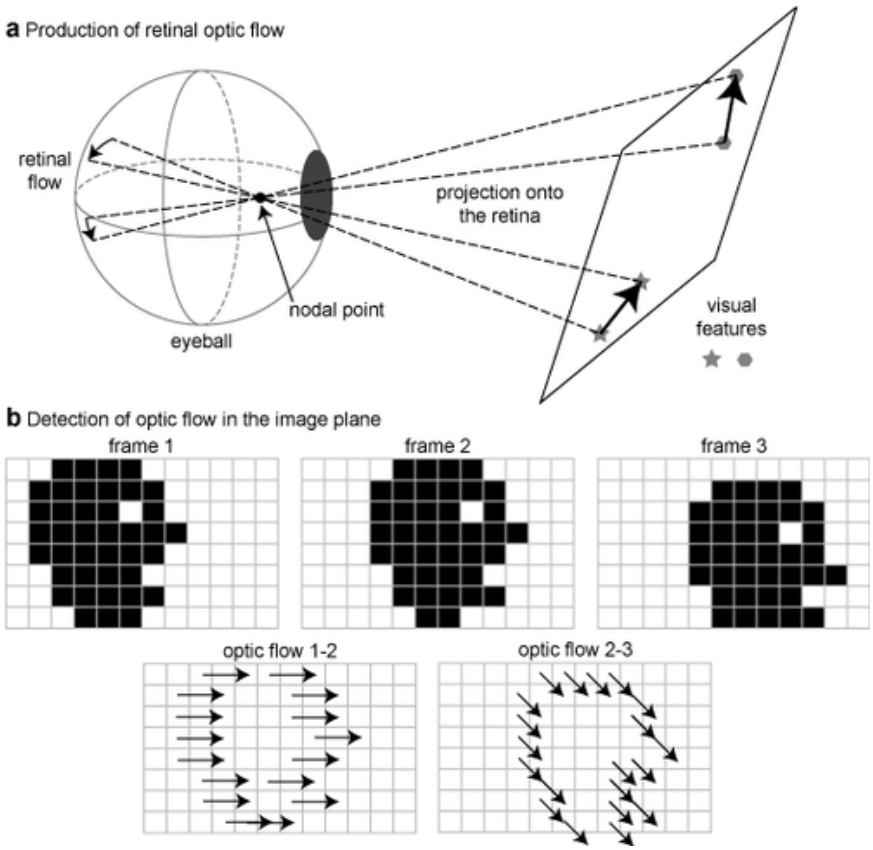
Problematyka wizyjnego pomiaru ruchu obiektów jest szeroko omawiana w literaturze. W [1] jest przedstawiony dwuetapowy proces wyznaczania ruchu na sekwencji obrazów w oparciu o „schematy intensywności" (ang. intensity-based schemes). Pierwszy etap obejmuje wykrywanie i pomiar ruchu poprzez przydzielenie wszystkim elementom obrazu kierunku i wartości wektora prędkości w oparciu o wyżej wymienione schematy intensywności. Drugi ma na celu podział sceny przedstawionej na obrazach na konkretne obiekty wyznaczenie ich trójwymiarowej struktury na podstawie wcześniejszych pomiarów prędkości. Algorytm w takiej postaci byłby jednak zbyt skomplikowany jak na nasze potrzeby – w założeniach uwzględniona została tylko sytuacja, w której jedynym przemieszczającym się elementem jest wentylator, a reszta obrazu jest statyczna. Poza tym, nie będziemy znajdować trójwymiarowej struktury wentylatora. Na szczęście została również w [1] opisana prostsza metoda wykrywania i pomiaru ruchu. Bazuje ona na porównaniu dwóch obrazów (O1 i O2 na rys. 1) z dwóch źródeł w punktach p_1 i p_2 rejestrujących ten sam ruch. Wyjście czujnika pierwszego w chwili t jest porównywane z wyjściem drugiego czujnika w chwili t – δt. Stąd też nazwa takiego schematu wykrywania ruchu: schemat opóźnionego

porównania (ang. delayed comparison scheme). Porównanie tu występujące może być realizowane na dwa sposoby. Pierwszy z nich mówi o mnożeniu wyjść obu czujników – jeśli obiekt poruszył się z punktu p_1 do punktu p_2 w czasie δt, to wynik takiego mnożenia będzie dodatni. Drugie zakłada, że jeden z obrazów zostanie zanegowany i dodany do drugiego. Dzięki temu odejmiemy tło, a na wynikowym obrazie zostanie tylko poruszający się element. Dwa źródła obrazów można zastąpić jednym – wtedy użyjemy dwóch następujących po sobie klatek – aczkolwiek będzie to wymagało założenia ruchu w dwóch wymiarach. Schemat opóźnionego porównania przy zastosowaniu jednego czujnika nie będzie potrafił zmierzyć ruchu w trójwymiarze. Taki system został zaprojektowany w oparciu o zasadę działania elementów wykrywających ruch w siatkówce królika (patrz [1]).

Inne podejście do zagadnienia pomiaru ruchu jest prezentowane w [2]. Tam autorzy opisują hierarchiczną strukturę przetwarzającą obrazy, na których jest wiele ruchomych punktów. Został tam również zaproponowany algorytm dopasowywania, który jest w stanie rozpoznawać i śledzić wszystkie ruchome obiekty. Posiadając 2 obrazy z jednego źródła, przesunięte w czasie, należy poddać je filtracji górno- i dolnoprzepustowej, a następnie podać wyjścia filtrów algorytmowi dopasowującemu. Należy go uruchomić dwukrotnie – raz, dając mu oba wyjścia filtrów górnoprzepustowych, a raz – dolnoprzepustowych. Dzięki temu, dostajemy dwa dopasowania, na podstawie których można określić ruch w różnych skalach. To podejście nie jest nam potrzebne i odrzuciliśmy jako zbyt skomplikowane.

Kolejną metodą, która jest często wykorzystywana do wyznaczania ruchu na podstawie sekwencji obrazów jest metoda przepływu optycznego (ang. optic/optical flow). Wyznacza ona zmiany naświetlenia elementów światłoczułych (np. siatkówka oka, czujnik w kamerze), które są konsekwencją ruchu względnego między elementem a jakimś obiektem. W literaturze występuje wiele definicji, zostanie tutaj przytoczona tylko jedna (podana w [5], wzięta z [10], tutaj we własnym tłumaczeniu na język polski): zauważalny ruch wzorców jasności obserwowany przy względnym ruchu kamery i śledzonego obiektu. Metoda ta polega na wyznaczaniu wektorów ruchu dla każdego bądź wybranych pikseli na sekwencji obrazów. W pierwszym przypadku mówimy o gęstym przepływie optycznym, w drugim – o rzadkim.

Istnieje wiele algorytmów realizujących oba te typy przepływu optycznego. Prawdziwy wysyp takich metod nastąpił w latach 80. – wtedy ukazały się takie algorytmy, jak: Horna – Schuncka, Lucasa – Kanade, Nagela czy Urasa. Z nowszych metod można wymienić algorytm Farnebäcka z 2000 r. Implementacje wszystkich są dostępne w [5].



Rysunek 2: Schemat powstawania przepływu optycznego (a) oraz przykładowy przepływ na sekwencji obrazów (b). Źródło: [5].

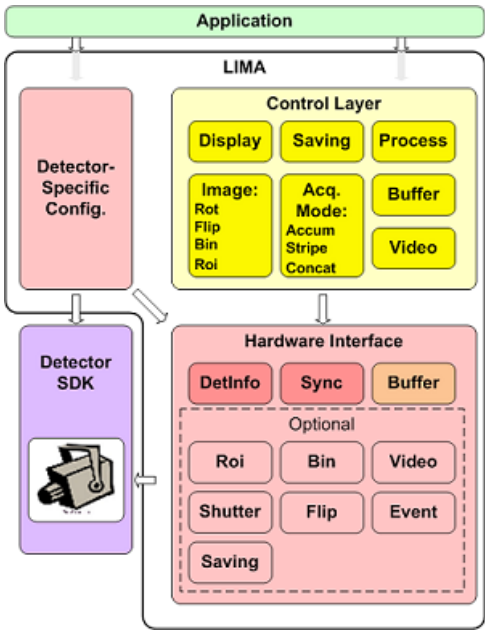
Pomimo, że wykrywanie obiektów ruchomych nie zostało szerzej omówione w [4], znalazł się tam jeden algorytm służący do wyznaczania ruchomych obiektów na podstawie dwóch obrazów - bieżącego i poprzedniego. Oba najpierw są poddawane działaniu filtra uśredniającego, aby zminimalizować szumy, a następnie binaryzacji. Następnie obejmuje się je od siebie, a na obrazie wynikowym wykonuje się operację otwarcia (erozja, a potem dylatacja) i podaje się go na algorytm wykrywający krawędzie. Jednocześnie zbinaryzowany obraz bieżący również podaje się na ten algorytm, a następnie poddaje się operacji dylatacji. Na tak otrzymanych dwóch obrazach wynikowych (różnicę po detekcji krawędzi oraz bieżący po dylatacji) wykonuje się koniunkcję logiczną. Na otrzymanym obrazie znajdują się tylko ślady ruchu obiektów śledzonych.

Taki algorytm wydaje się być najbardziej odpowiedni do naszych zastosowań. Umożliwia wykorzystanie jednego źródła obrazów, które wykonuje akwizycję z zadaną częstotliwością, a przy tym jest na tyle nieskomplikowany, aby nie trzeba było stosować żadnego dedykowanego sprzętu do obliczeń.

Rynek oprogramowania do sterowania kamerami jest pełny drogich, specjalistycznych programów. Na szczęście istnieją również rozwiązania darmowe, udostępniane na zasadach otwartego oprogramowania. Jednym z nich jest biblioteka LImA (ang. Library for Image Acquisition) – projekt opracowany w ESRF (European Synchrotron Radiation Facility), który ma na celu unifikację sterowania dwuwymiarowymi czujnikami wizyjnymi (jego strona internetowa jest podana w [11]). Wszystkie one posiadają wspólne cechy i obszary zastosowań, ale zwykle dostarczane są z dedykowanym oprogramowaniem, co jest problematyczne, jeśli zachodzi potrzeba używania urządzeń od więcej niż jednego producenta. Projekt LimA ma 3 główne cele:

1. Zapewnić niezależność interfejsu prezentowanego systemowi sterowania od interfejsu samego urządzenia.
2. Dać użytkownikom duży zasób funkcjonalności programowych, które nie są obsługiwane przez sprzęt.
3. Wykorzystać wielowątkowe algorytmy przetwarzania synchronizowane komunikacją zdarzeniową, aby być w stanie przetworzyć strumienie danych z czujników.

Pierwszy cel jest realizowany przez zastosowanie tego oprogramowania w systemie sterowania Tango, również wymyślonym przez ESRF, a obecnie utrzymywanym przez konsorcjum synchrotronów z całej Europy. Jest to zestaw narzędzi do sterowania rozproszonymi systemami sterowania, wykorzystywany głównie w dużych placówkach fizycznych.



Rysunek 3: Struktura pakietu LimA. Źródło: [11].

Biblioteka ma budowę modułową (jej schemat ideowy jest widoczny na rys. 3). Składa się z dwóch warstw – programowej (sterowania), która zawiera moduły odpowiadające za przetwarzanie i analizę obrazów, jak również zarządzające odpowiadaniem na zapytania użytkowników oraz sprzętową, która komunikuje się z konkretnym urządzeniem i przechwytuje z niego obraz.

3. Analiza złożoności i estymacja zapotrzebowania na zasoby

Sformułowanego wcześniej problem posiada konkretne zapotrzebowanie na różne parametry obu części. Głównym wymaganiem dla części programowej jest duża prędkość przetwarzania i analizy szybko zmiennych obrazów. W szczególności, powinien być w stanie przyjmować strumień danych z kamery oraz obrabiać go z wystarczającą prędkością. Jeśli chodzi o układ wyzwalający, to kluczowa jest minimalizacja czasu reakcji wszystkich modułów zastosowanych do detekcji ruchu.

Obiekt szybkozmienny powinien posiadać możliwość regulacji swojej prędkości w z sposób łatwy, dokładny a jednocześnie bez przerwania aktualnie prowadzonego badania. Zmiany prędkości mogą mieć częstotliwość kilkunastu - kilkudziesięciu Hz. Preferowane jest sprzętowa realizacja kontroli prędkości, najlepiej z bezpośrednim udziałem osoby prowadzącej eksperyment. Zakładamy, że obiekt ten będzie miał prędkość regulowaną w zakresie od około 500 obr./min do około 2000 obr./min.

W części sprzętowej należy stworzyć układ redundantny, który ma możliwość mierzenia prędkości obrotowej obiektu z jak najmniejszymi opóźnieniami oraz z dużą odpornością na zakłócenia. W związku z założeniem prędkości akwizycji rzędu kilku kHz, opóźnienia pojawiające się w tym układzie nie powinny być większe niż kilkadziesiąt milisekund. Dodatkowo ma on pełnić rolę układu wyzwalającego akwizycję obrazu w związku z czym powinien móc komunikować się z kamerą w standardzie uprzednio wybranego złącza kamery.

Układ analizujący rozpoczęcie ruchu obiektu poprzez analizę sygnału podawanego przez układ akwizycji powinien wprowadzać jak najmniejsze opóźnienia oraz posiadać zaimplementowany sprzętowo lub programowo algorytm detekcji tylko prawdziwych sygnałów wyzwalających niwelując zakłócenia oraz szumy. Będą one najprawdopodobniej miały charakter zmiennego poziomu oświetlenia na stanowisku pomiarowym. Przewidujemy konieczność zastosowania dodatkowego, dedykowanego źródła oświetlenia. Powinno ono być wystarczająco jasne, aby wykluczyć wszelkie zakłócenia natury świetlnej, mogące występować w laboratorium. Dodatkowo zadaniem układu wyzwalającego jest wysłanie sygnału rozpoczęcia akwizycji obrazu z odpowiednią częstotliwością zadaną przez użytkownika.

Urządzenie odpowiedzialne za rejestrację obrazu w sposób cyfrowy powinno się charakteryzować możliwością akwizycji odpowiedniej ilość klatek na sekundę. Szybkość rejestracji powiązana jest przede wszystkim z wysoką jasnością obiektywu, małego opóźnienia migawki oraz dużą szybkością przesyłu danych. Przy założeniu iż stanowisko badawcze jest elementem stacjonarnym ważnymi parametrami są czułość oraz rozdzielczość. Dodatkowo, urządzenie rejestrujące musi mieć możliwość zdalnego, sprzętowego wysyłania mu sygnału wyzwalającego, częstotliwości akwizycji. Musi również posiadać złącze do szybkiej transmisji danych, niezależne od złącza służącego do przesyłania sygnału wyzwolenia.

Algorytm odpowiedzialny za przetwarzanie powinien pozwolić na śledzenie ruchu obiektu przy częstotliwości obrotów rzędu kilkuset Hz do 2 kHz. W wykorzystywanym algorytmie detekcji ruchu zakłada się iż obiekt rejestrujący jest statyczny - porusza się jedynie obiekt na analizowanym obrazie. Obliczenia powinny być prowadzony w sposób ciągły, a czas potrzebny na przetworzenie jednego obrazu powinien być krótszy niż na czas potrzebny na jego wykonanie. Pozwoli to na rejestrację zmian w czasie rzeczywistym. Realizowany algorytm powinien wymuszać regularne pobieranie klatek obrazu prosto z kamery w postaci strumienia danych. Dodatkowo, tak zrealizowany pomiar prędkości powinien również posiadać większy zakres niż obiekt dla którego to będzie zastosowany.

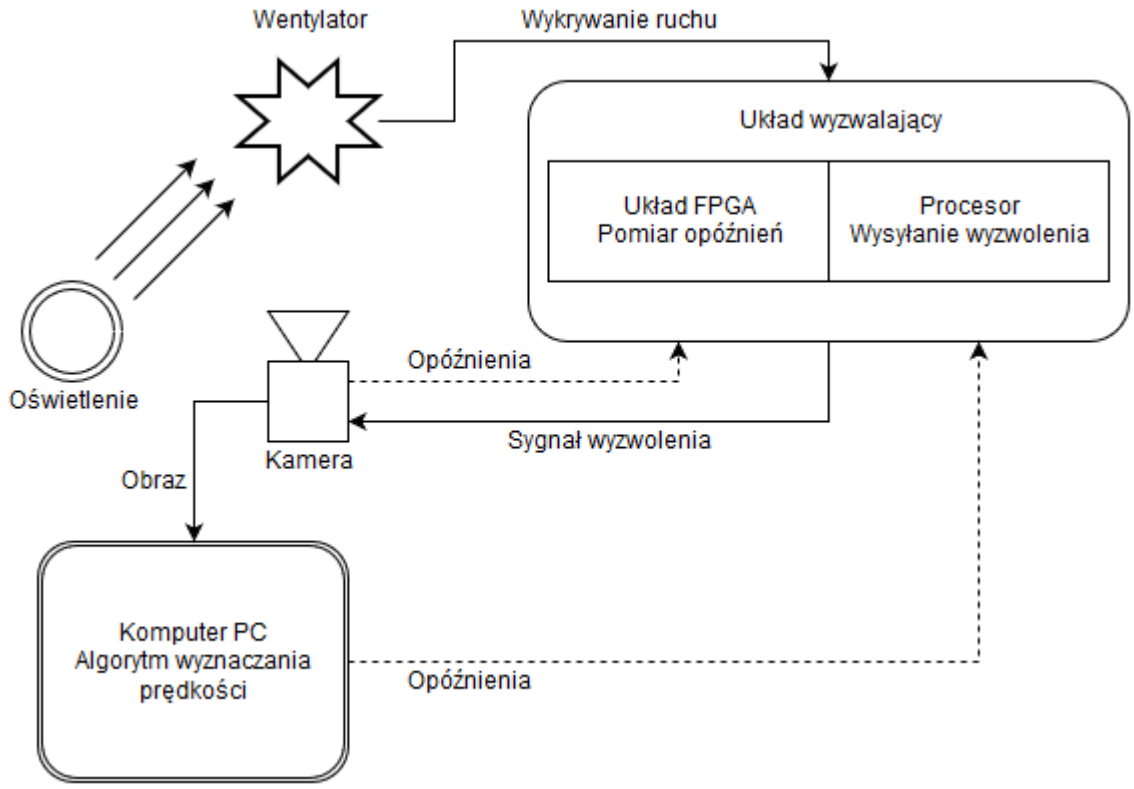
Część sprzętowa, której zadaniem będzie pomiar wszystkich opóźnień wymienionych w rozdziale 2, powinna być w stanie odbierać i mierzyć te opóźnienia. Z uwagi na małą ilość tych sygnałów, nie będzie potrzebna duża ilość zasobów logiki rekonfigurowalnej, a więc nie wykonaliśmy dokładniejszej analizy potrzebnych układów.

4. Koncepcja proponowanego rozwiązania

Ogólny schemat proponowanej koncepcji rozwiązania problemu nakreślonego w poprzednich rozdziałach jest przedstawiony na rys. 4.

Jako obiekt szybko zmienny wykorzystany zostanie wentylator komputerowy zasilany prądem stałym o zakresie napięciowym 6 - 12 V. Spełnia on bowiem podstawowe założenia jakimi się kierowaliśmy: cena oraz odporności na błędy użytkownika. Wymusza ona prawidłowe podłączenie zasilania: z racji możliwości obrotu tylko w jedną stronę, przyłożenie napięcia o nieprawidłowej biegunowości skutkować będzie nadmiernym grzaniem się układu co może prowadzić do spalenia urządzenia. Jest to jednak proces długotrwały i zależy ściśle od wartości przyłożonego napięcia. Taki dobór obiektu umożliwia również zmontowanie układu realizującego regulację prędkości poprzez regulację napięcia wysyłanego bezpośrednio do tego wentylatora. Na

wentylatorze umieścimy znacznik, którego obecność będziemy badać i na tej podstawie wyznaczymy moment wykonania przez wentylator jednego obrotu.



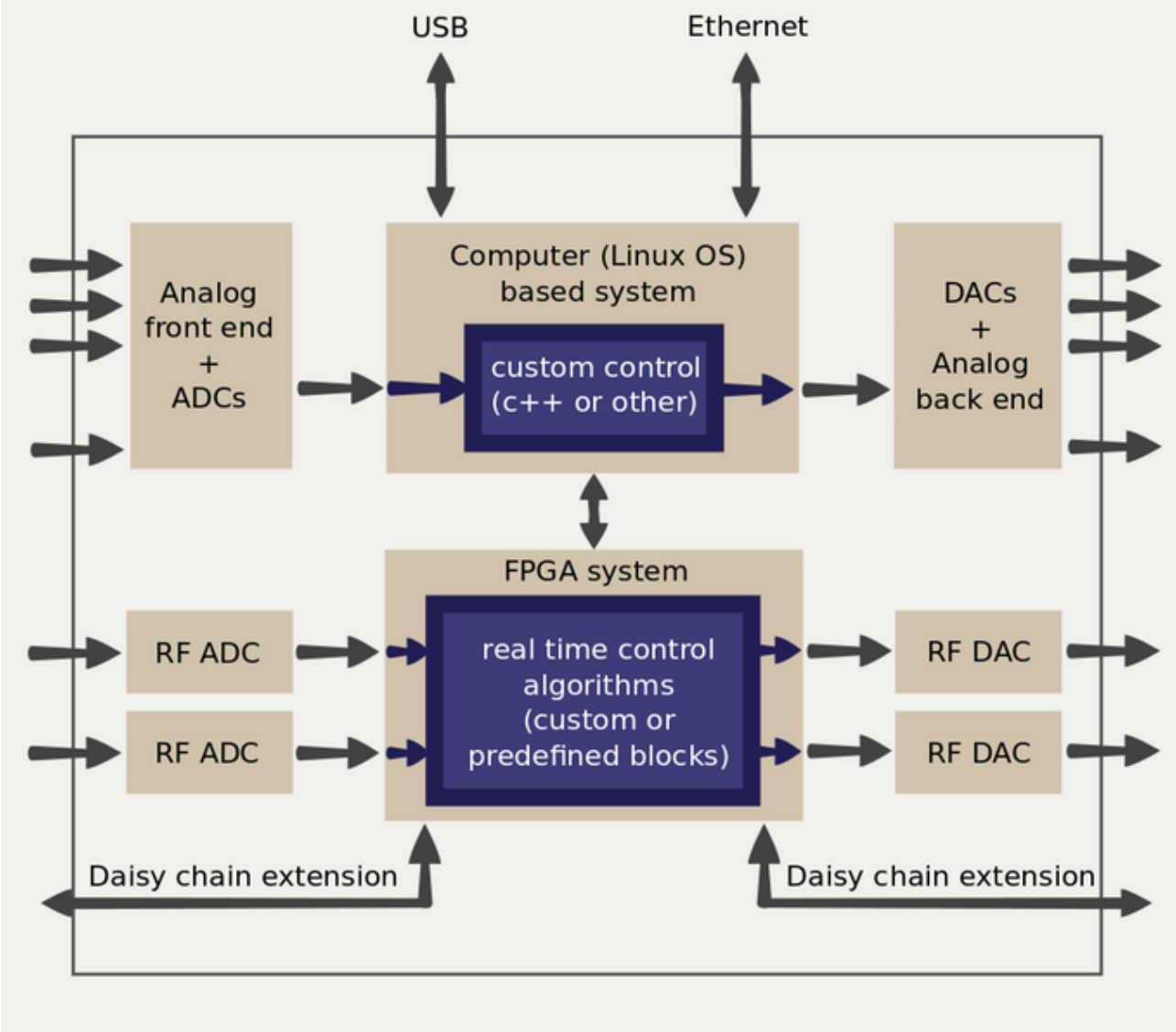
Rysunek 4: Schemat całości systemu pomiarowego.

W celu rejestracji jego prędkości obrotowej stworzone zostaną dwa moduły sprzętowe. Pierwszy to układ elektryczny, który nie będzie wykorzystywał elementów cyfrowych – jego budowa ma być nieskomplikowana, przez co minimalizować będzie szansę błędu. Niestety, z racji swojej prostoty nie będzie posiadał zbyt dużej uniwersalności i będzie mógł być wykorzystywany tylko dla przyjętego obiektu szybko zmiennego lub niewiele się różniącego. Do wykrywania ruchu obiektu będzie wykorzystywać półprzewodnikowy element optoelektroniczny - transoptor. Z racji próby minimalizacji zakłóceń wykorzystywany element będzie w obudowie zamkniętej ze szczeliną o szerokości większej od szerokości wentylatora komputerowego. Układ ten powinien posiadać szereg wyjść testowych w celu wyprowadzenia wszystkich potrzebnych sygnałów. Powinien mieć możliwość dopasowywania większość swoich parametrów do potrzeb użytkownika - w tym celu zastosowane zostaną odpowiednie potencjometry. Prędkość badanego obiektu będzie można oszacować wykorzystując w tym celu oscyloskop podpięty na wyjście modułu oraz wybranie trybu obserwacji częstotliwości sygnału wyjściowego. W ten sposób uzyskana zostanie ilość łopatek wentylatora przypadająca na daną jednostkę czasu. Znają liczbę łopatek w danym wentylatorze możemy oszacować chwilową prędkość obrotu z dużą dokładnością.

Dobrana przez prowadzącego kamera to Basler acA2000-165uc. Jest to przemysłowy sprzęt wysokiej jakości - umożliwia ona akwizycję 165 ramek na sekundę przy maksymalnej rozdzielczości, wynoszącej 2 Mpx. Oczywiście, nie bedziemy potrzebować aż takiej rozdzielczości - wystarczą nam obrazki rozdzielczości około 100 x 100 px. Dzięki temu będziemy mogli zwiększyć prędkość akwizycji oraz pozbędziemy się z obrazów niepotrzebnych elementów otoczenia - istotne jest jedynie, aby kamera mogła "zobaczyć" jeden fragment znacznika. To urządzenie posiada dwa złącza. Pierwsze jest optoizolowaną wtyczką służącą do przesyłu sygnału wyzwalającego. Interfejsem, który jest wykorzystywany do wysyłania ramek obrazu jest USB 3.0. Dodatkową zaletą tego sprzętu jest dwojaka możliwość konfiguracji – można w tym celu użyć dedykowanego oprogramowania (Pylon, w laboratorium obecna wersja 4) lub stworzyć własny projekt, który będzie wykorzystywał dostarczone przez producenta SDK (Pylon SDK, również w wersji 4). Wszystkie opisane funkcje tej kamery sprawiają, że jest ona wystarczająca do naszych potrzeb.

Drugi moduł sprzętowy bazować będzie na płytce RedPitaya i będzie miał trzy funkcje: przyjęcie sygnału wykrytego ruchu, wysyłanie sygnału wyzwalającego oraz pomiar opóźnień. Wspomniany system zawiera układ SoC Xilinx Zynq 7010, w którego skład wchodzi dwurdzeniowy procesor w architekturze ARM (Cortex - A9) oraz układ FPGA. Został on wzbogacony o szereg interfejsów, m.in. szybkie wejścia i wyjścia oscyloskopowe, 12-bitowe przetworniki analogowo-cyfrowe, 2 złącza typu SATA, Ethernet oraz wejście na kartę microSD. Dokumentacja jest dostępna pod adresem wymienionym w [7], wszelkie dodatkowe informacje w [6], a schemat ideowy jest widoczny na rys. 5. Planujemy użyć części procesorowej do detekcji ruchu, a więc odbioru sygnału z wyżej wymienionego transoptora, a części rekonfigurowalnej do mierzenia opóźnień. Wybór padł właśnie na ten układ, ponieważ - w odróżnieniu od platform obecnych w laboratorium - mogliśmy korzystać z niego poza uczelnią. Poza tym, istnieje duża liczba darmowych, gotowych do użycia aplikacji, które mogą być pomocne w rozwiązywaniu zadań projektowych.

Opracowanie programów na obie części tego układu będzie się odbywać w dwóch środowiskach programistycznych. W przypadku procesora, będzie to program Eclipse z wtyczką umożliwiającą pisanie programów w języku C++. Będzie do tego potrzebny kompilator oraz zestaw bibliotek - zdecydowaliśmy się użyć dystrybucji Cygwin, a biblioteki potrzebne do obsługi konkretnego procesora na płytce RedPitaya pobrać z oficjalnego repozytorium tego projektu (można jego adres znaleźć w [6]). Jeśli chodzi o układ FPGA, to użyjemy programu ISE WebPack firmy Xilinx.



Rysunek 5: Schemat ideowy układu RedPitaya. Źródło: [6].

Połączenie tego modułu z układem wykrywającym ruch będzie mogło być zrealizowane dwojako. Przede wszystkim, można użyć przetwornika Schmitta, aby wygładzić wszelkie niedoskonałości sygnału z transoptora i podać go na wejście cyfrowe wybranej płytki. Oczywistą zaletą takiego rozwiązania jest prosty sposób obróbki takiego sygnału. Drugi sposób to użycie jednego z analogowych wejść oscyloskopowych w układzie RedPitaya. Zaletą drugiego rozwiązania jest to, że będziemy mogli przeprowadzić analizę częstotliwościową bezpośrednio na platformie sprzętowej, dzięki czemu uzyskamy referencyjny pomiar prędkości obrotowej wentylatora. Niestety, ten sposób posiada również oczywistą wadę - tor analogowy będzie wprowadzał dodatkowe opóźnienia w całym systemie. Niezależnie od sposobu podłączenia sygnału

z transoptora do płytki, algorytm wykrycia ruchu będzie polegał na detekcji zbocza narastającego tego sygnału i wystawieniu sygnału PWM, który będzie miał na celu wyzwolić akwizycję z kamery.

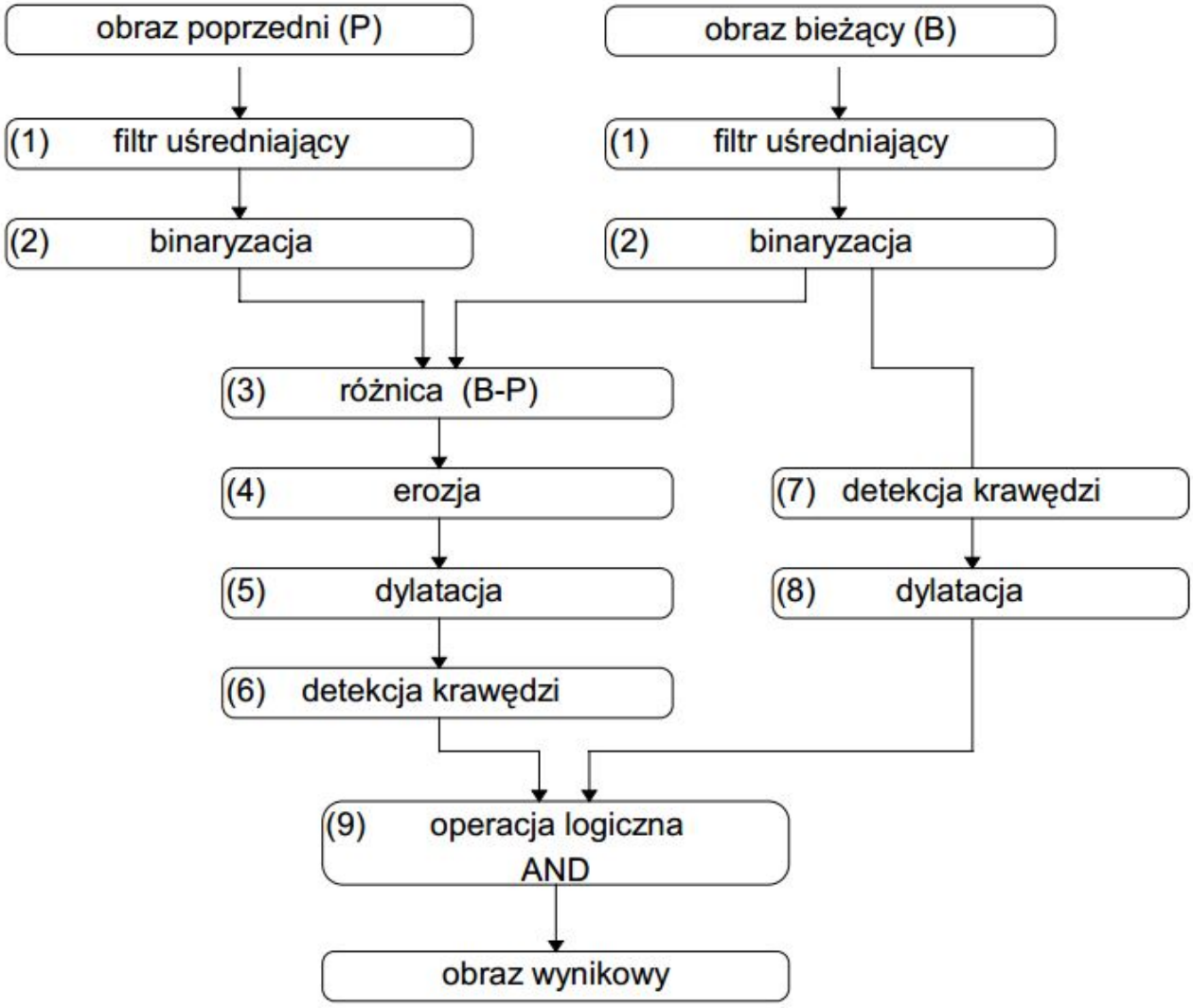
Część programowa będzie oparta o bibliotekę akceleracji obliczeń OpenCL. Wykorzystanie zasobów karty graficznej powinno dać nam wystarczającą prędkość przetwarzania obrazów, aby przetworzyć dane z wymienioną w poprzednim rozdziale częstotliwością akwizycji. Oprogramowanie powinno się składać z trzech części: połączenia z kamerą i wysłania jej odpowiednich ustawień, sprawdzania obecności znacznika oraz wyznaczania prędkości obrotowej na podstawie ilości ramek, które zostały przetworzone od ostatniej wykrytej pozycji znacznika.

Pierwsza część będzie zrealizowana z wykorzystaniem bibliotek Pylon SDK, dostarczonych przez firmę Basler, producenta kamery. Umożliwi to wysłanie kamerze uprzednio przygotowanego pliku z ustawieniami oraz bezpośrednie pobieranie strumienia danych wejściowych. Część druga - algorytm wyznaczania obecności znacznika – został zaprezentowany na rys. 6. Jego opis tekstowy został zamieszczony we wstępie, ponieważ jest to algorytm z [4].

Część trzecia będzie przechowywać numer ramki, na której został poprzednio zaobserwowany znacznik, a następnie przy kolejnym napotkaniu znacznika wyznaczy wartość prędkości wentylatora według wzoru 1.

$$\begin{aligned} \frac{f_{akwizycji}}{i_{bieżące} - i_{poprzednie}} &= f_{obrotu} \\ v_{wentylatora} &= f_{obrotu} * 60 \end{aligned} \tag{1}$$

Indeksy ramek bieżącej i poprzedniej są opisane symbolem i. Częstotliwości powinny być podane w Hz, a prędkość wentylatora wychodzi w obr./min.

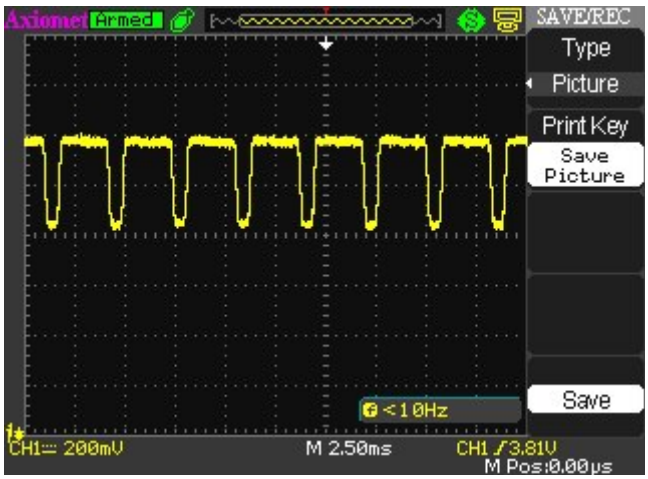


Rysunek 6: Schemat blokowy algorytmu wykrywania ruchu. Źródło: [4].

5. Symulacja i Testowanie

Modelowanie i symulacja

W związku z tym, że nie używaliśmy żadnej platformy rekonfigurowalnej w ostatecznej wersji projektu, nie przeprowadzaliśmy również żadnych procedur symulacyjnych.

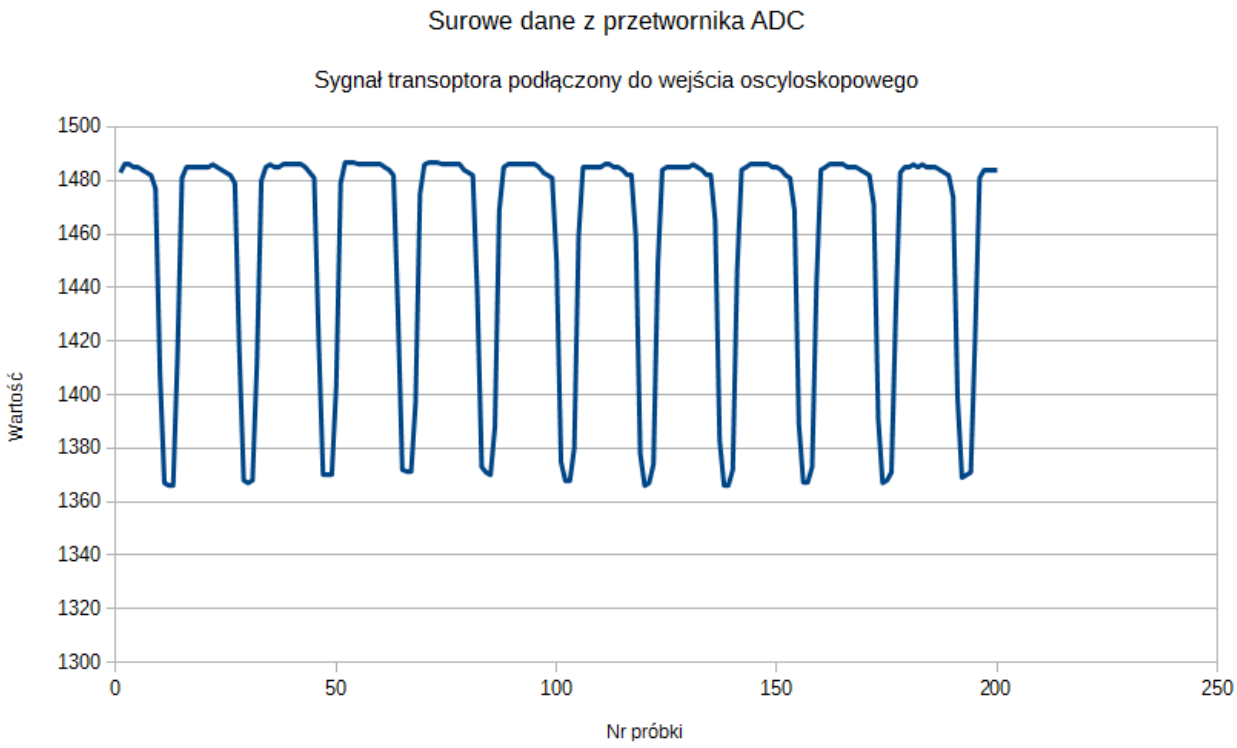


Rysunek 7. Pomiary napięcia na transoptorze w czasie ruchu wentylatora.

Testowanie i weryfikacja

Działanie obu części projektu zostało przetestowane na kilka sposobów. Przede wszystkim, część sprzętowa została złożona, a następnie przeprowadziliśmy procedurę weryfikacji połączeń przy pomocy multimetru. Po wprowadzeniu koniecznych poprawek, sprawdziliśmy jakość sygnału otrzymywanego z transoptora oraz odpowiednio dostroiliśmy sprzęt, tak aby spełniał nasze założenia (jakość sygnały, szerokość strefy histerezy). Na rys. 7 jest przedstawiony obraz uzyskany na oscyloskopie w czasie ruchu wiatraczka. Jedna podziałka na osi pionowej odpowiada wartości 200 mV, a na osi poziomej – 2.5 ms. W związku z tym możemy określić 2 parametry funkcjonowania układu. Przede wszystkim, napięcie transoptora zmienia się w zakresie 3,8 V do 4,2 V – jest on nieprzyjemny w obróbce, bo spodziewaliśmy się sygnału odpowiadającego poziomom napięcia standardu TTL. Po drugie, prędkość obrotowa wentylatora w tej krótkiej próbce wynosiła około 45 obr./s, czyli 2700 obr./min.

Sygnał z transoptora został również zmierzony po podaniu go na wejście oscyloskopowe płytki RedPitaya. Z racji możliwości manualnego dobrania wzmocnienia przy wykorzystaniu zworki, wstępny zakres został ustawiony na +-20 V, cały sygnał otrzymywany z czujnika mieścił się w zakresie. Rozdzielczość przetwornika dobrano tak, aby zajmowała 12 bitów czyli przetwornik przetwarzał sygnał na jedną z 4096 wartości liczbowych. Otrzymana w ten sposób rozdzielczość napięcia równa była 9.25 mV. Zgodnie ze specyfikacją producenta maksymalna szybkość próbkowania dla takiej rozdzielczości to 120 kS/s czyli ilość pomiarów na sekundę. W programie w celu przyśpieszania obliczeń obcięliśmy zakres pomiarowy do interesujących nas wartości czyli 0 - 5 V. Wykres z tego testu jest pokazany na rys. 8.



Rysunek 8: Wykres surowych danych z przetwornika ADC mierzących sygnał z transoptora.

Układ wyzwalający w pierwotnej wersji miał wykorzystywać komparator napięć. Niestety z racji zbyt małego prądu podawanego przez transoptor (800 µA) nie było możliwości poprawnego wykorzystanie tego elementu, nie potrafił on zmierzyć napięcia podawanego przez transoptor. Kolejnym problemem były zbyt mały prąd na wyjściu komparatora wynoszący zgodne ze specyfikacją tylko 20 mA. Z racji braku czasu potrzebnego na zamówienie odpowiedniego układu scalonego jakim jest LM393 wykorzystano płytkę Arduino z mikrokontrolerem ATmega 328 i zaimplementowano na niej prosty komparator z możliwością wyświetlania danych na przyłączonym i skonfigurowanym wyświetlaczu. Napięcia referencyjne, jak i z wyjścia transoptora zgadzały się z założonymi, a układ poprawnie wyzwalał zrobienie zdjęcia. Jedynym problemem była niezgadzające się szybkość ramek wynosząca maksymalnie 700 FPS.

W trakcie implementacji program przeprowadziliśmy szereg testów mających na celu jak najlepszą detekcje znacznika oraz wykrycie ewentualnych błędów. Polegały one na dopasowywaniu progów binaryzacji, wyświetleniu kluczowych pod elementów programu oraz mierzeniu czasu poszczególnych modułów. W trakcie testów doszliśmy do wniosku, że operacje na zdjęcie kolorowym, czyli zawierającym 3 bajty na piksel pochłania zbyt dużo zasobów i lepiej wykonać wstępna na obraz posiadający tylko skalę szarości.

6. Rezultaty i wnioski

Finalnym rozwiązaniem jest seria modułów realizujących różne funkcjonalności. Wymienione zostaną w kolejności w jakiej działają. Komponent rejestrujący ruch obiektu bazuje na transoptorze zgodnie z założeniami, podłączona pod niego płytka z ATmegą ma za zadanie proste wyświetlanie pobieranego sygnału oraz realizację funkcjonalności układu porównującego dwa napięcia. W zależności od porównywanych sygnałów zostanie wystawione logiczne “zero” lub “jeden”. Sygnał ten zostanie przepuszczony przez wzmacniacz prądowy bazujący na tranzystorze BC547. Tak wzmocniony sygnał traktowany jest jako zasilanie dla płytki sterującej wyzwalaniem kamery (jest to nie nasz projekt, dlatego taki sposób sterowania najmniej ingeruje w działanie tego elementu). Spadek zasilania traktowany był przez kamerę jako sygnał wyzwalający.

Wykorzystując program Pylon ustawiliśmy potrzebne zmienne związane z trybem wyzwolenia kamery. Wykorzystaliśmy tryb “FrameBurst”. Z racji problemów z bezpośrednim pobieraniem strumienia obrazów do aplikacji analizującej zapisywane były one pliku. Dlatego też jednorazowo zapisywaliśmy kilkaset obrazów. Każdy z nim zrobiony był w momencie podania sygnału wyzwolenia poprzez odcięcie zasilania płytki.

Tak zapisane obrazy analizowaliśmy przy wykorzystaniu specjalnej aplikacji napisanej w OpenCV i realizującej algorytm wykrywania znacznika (rys. 3), oraz liczącej aktualną prędkość kątową.

Aktualna prędkość pomiarowa ograniczona jest przez wcześniej wspomnianą płytkę i wynosi 700 fps co daje około 6-10 klatek na obrót wentylatora zasilanego napięciem 6 V. Prędkość przetwarzania jednej ramki o szerokości 96 pikseli i wysokości 50 wynosi 400 μs. W celu przyśpieszenia rejestracji obrazów należałoby użyć układu scalonego realizującego porównanie sygnałów.

7. Podsumowanie

Za nasze największe osiągnięcia uważamy szybki algorytm do przetwarzania obrazów oraz układ konwertujący obrót wentylatora na sygnał analogowy o modulowanym zakresie. Postawiony temat był bardzo ciekawy, a jednocześnie wymagający. Nauczyliśmy się podstaw związanych z lutowaniem oraz analizy specyfikacji technicznych. Do tego czasu niestety nie mieliśmy możliwości budowania stanowiska badawczego od podstaw, a jedynie wykorzystywaliśmy gotowe. Natknęliśmy się na sporo trudności z tym związanych jak na przykład prawidłowe rozmieszczenie elementów podczas montażu, gdzie każdy błąd przez nas popełniony wiązał się tak naprawdę z kolejną wersją całego układu. Wiedza nabyta podczas obsługi camery Baslera będzie bardzo przydatna w naszej dalszej pracy, gdyż mamy z nią styczność prawie na co dzień. Implementacja

poszczególnych typów przekształceń morfologicznych pozwoliła odświeżyć informacje związane z przetwarzaniem obrazu oraz zobaczyć efekty działania kilku tych przekształceń naraz.

Niestety jak zostało wspomniane wcześniej nie udało się dokończyć projektu. Naszym zdaniem należałoby zmodyfikować układ pomiarowy tak aby cała jego funkcjonalność mieściła się na jednej płytce, oraz pozwalała na łatwy dostęp do wszystkich sygnałów w celach testowych. Program do analizy obrazy powinien być niestety przepisany do C++ gdyż daje on możliwość łatwiejszego powiązania kamery z programem i bezpośredni przepływ informacji bez potrzeby ich zapisywania.

8. Literatura

[1] Hildreth, E. C., Ullman S., "The Measurement of Visual Motion", MIT Artificial Intelligence Laboratory, 1982.

[2] Anadan, P., „A Computational Framework and an Algorithm for the Measurement of Visual Motion", International Journal of Computer Vision, numer 2, strony 283-310, 1989.

[3] Barron, J. L., Fleet, D. J., Beauchemin, S. S., „Performance of Optical Flow Techniques", International Journal of Computer Vision, numer 12:1, strony 43-77, 1994.

[4] Tadeusiewicz R., Korohoda P., „Komputerowa analiza i przetwarzanie obrazów." Wyd. Fundacji Postępu Telekomunikacji, Kraków, 1997.

[5] Optic flow, http://www.scholarpedia.org/article/Optic_flow

[6] Projekt w serwisie Kickstarter – RedPitaya. https://www.kickstarter.com/projects/652945597/red-pitaya-open-instruments-for-everyone/description

[7] Specyfikacja układu RedPitaya. https://www.dropbox.com/s/b2pxkwj1ljzyw71/Red_Pitaya_HW_Specs_V1.1.1.pdf

[8]Specyfikacja techniczna transoptora KTIR0911S http://www.tme.eu/en/Document/92b21f3f26b6948910585b1e34d6ef32/KTIR0911S.pdf

[9] Dokumentacja kamery Basler ace2000 – 165uc. http://www.baslerweb.com/en/products/area-scan-cameras/ace/aca2000-165uc

[10] Horn, B.K.P. "Robot vision". MIT Press, 1986.

[11] Strona internetowa projektu LimA. http://lima.blissgarden.org/index.html

9. DODATEK A: Szczegółowy opis zadania

Specyfikacja projektu

Tematem projektu było stworzenie stanowiska badawczego służącego do wyznaczania prędkości obrotowej obiektów. Układ powinien automatycznie uruchamiać tor przetwarzania w tym kamerę po wykryciu ruchu. W trakcie działania powinny być mierzone opóźnienia w przesyle danych między głównymi podzespołami. Oprogramowanie miało wykorzystywać akceleracją obliczeń biblioteką OpenCV lub OpenCL. Należy wykorzystać kamerę Basler acA2000-165uc. Wszelkie dodatkowe informacje zostały zawarte w rozdziałach 2 i 3.

Szczegółowy opis realizowanych algorytmów przetwarzania danych.

W niniejszym projekcie występuje jeden algorytm przetwarzania danych, realizujący detekcję ruchu na sekwencji obrazów poprzez rozpoznawanie wskaźnika. Wzorowaliśmy się na pracy [4]. W skład zaimplementowanego algorytmu detekcji obrazów ruchomych wchodzi kilka procedur przetwarzania obrazów. Wszystkie funkcje zostały zaimplementowane w bibliotece OpenCV z której korzystaliśmy i nieznacznie modyfikowaliśmy w celu przystosowania do naszego algorytmu. Z racji wykorzystywanego języka oraz komputera z systemem Windows wszystko przebiega sekwencyjnie i nie wykorzystywane, są żadne elementy wielowątkowości lub asynchroniczności.

Obrazy w celu zmniejszenia złożoności obliczeniowej zostają konwertowane do skali szerokości. Pozwala to zaoszczędzić 2 bajty na piksel. Jest to przeprowadzenie z racji zachowywania poszczególnych etapów operacji w pamięci. Po zmniejszeniu rozmiary macierz informacji zostaje podana wstępnej filtracji poprzez zastosowanie filtra uśredniającego. Ma to na celu minimalizację szumów które pojawiają się z racji dużego wzmocnienia kamery. Działanie tego filtra opisuje wzór 2. Funkcja h(k, l) opisuje element strukturalny – w naszym przypadku macierz samych „1” o wymiarach 5 x 5.

$$g(i,j)=\sum_{k,l} f(i+k,j+l)*h(k,l) \tag{2}$$

Na tak przefiltrowanych obrazach przeprowadzania jest binaryzacja. Wykorzystany próg jest tak dobrany, aby tylko obrazy zawierające znacznik posiadały jakiekolwiek białe elementy. Wykorzystując poprzedni obraz który został zachowany w pamięci w celu uniknięcia powtarzania początkowych etapów wykonuje się odejmowanie poprzedniego od aktualnego. Obraz wynikowy należy następnie poddać otwarciu mającym na celu usunięcie pojedynczych pikseli różnych od otoczenia. W obu tych przekształceniach używamy elementu strukturalnego o postaci macierzy samych „1” o wymiarach 5 x 5. Kolejnym krokiem jest przeprowadzenie wykrywania krawędzi. W

tym celu wykorzystujemy algorytm Canny'ego. Jego procedura może być opisana w następujący sposób:

1. Usuń szumy filtracją Gaussa. Wykorzystuje on wzór 2 z elementem strukturalnym postaci:

$$k=\frac{1}{159}*\begin{bmatrix}2&4&5&4&2\\4&9&12&9&4\\5&12&15&12&5\\4&9&12&9&4\\2&4&5&4&2\end{bmatrix}$$

2. Znajdź gradient intensywności na obrazie:
   1. Wykorzystaj dwie maski konwolucji (w kierunku x i y):

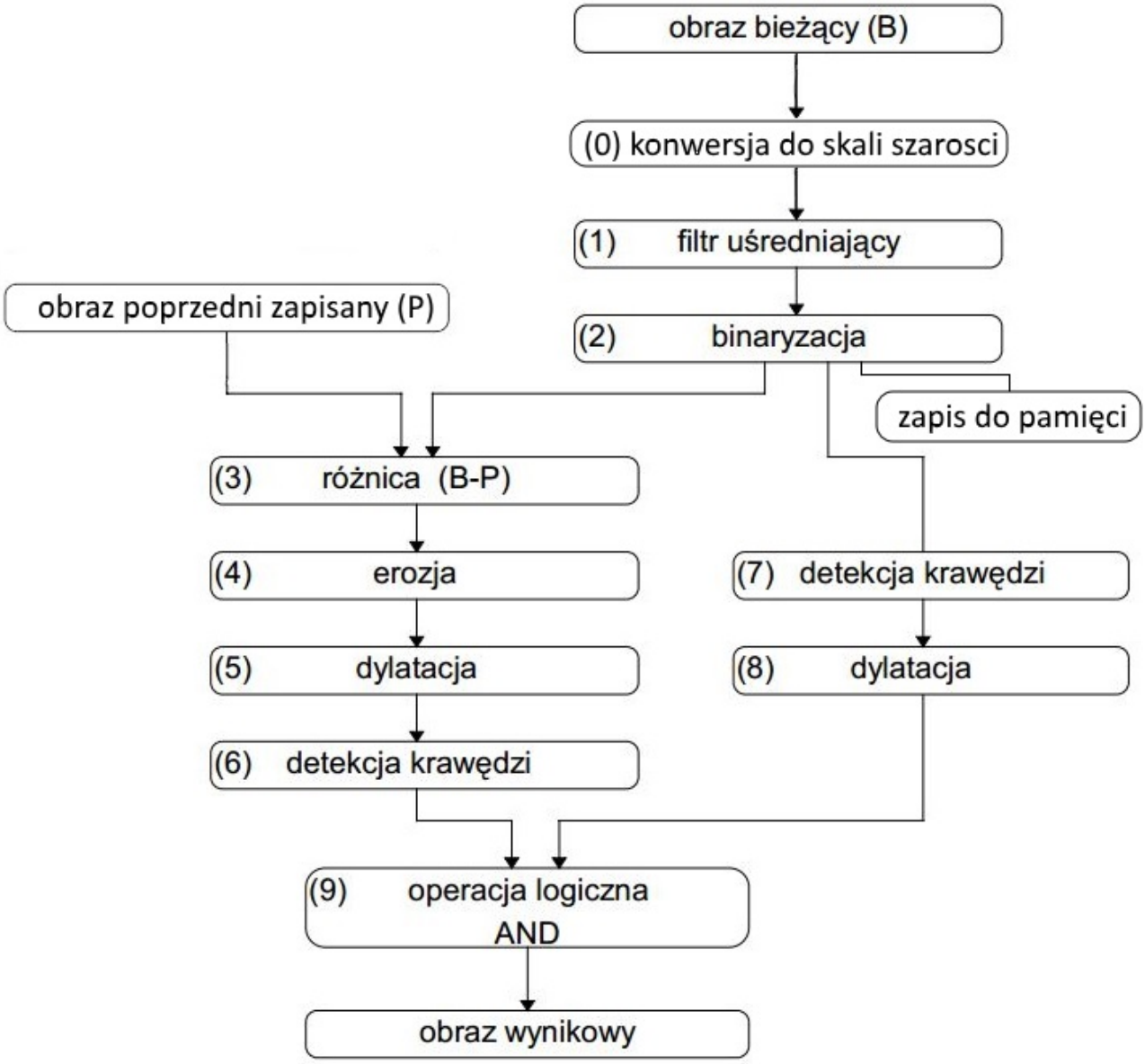
$$G_x=\begin{bmatrix}-1&0&1\\-2&0&2\\-1&0&1\end{bmatrix},\quad G_y=G_x^T$$

   2. Znajdź długość i kierunek wektora gradientu według wzoru 3.

$$G=\sqrt{G_x^2+G_y^2}$$
$$\theta=\arctg\left(\frac{G_y}{G_x}\right) \tag{3}$$

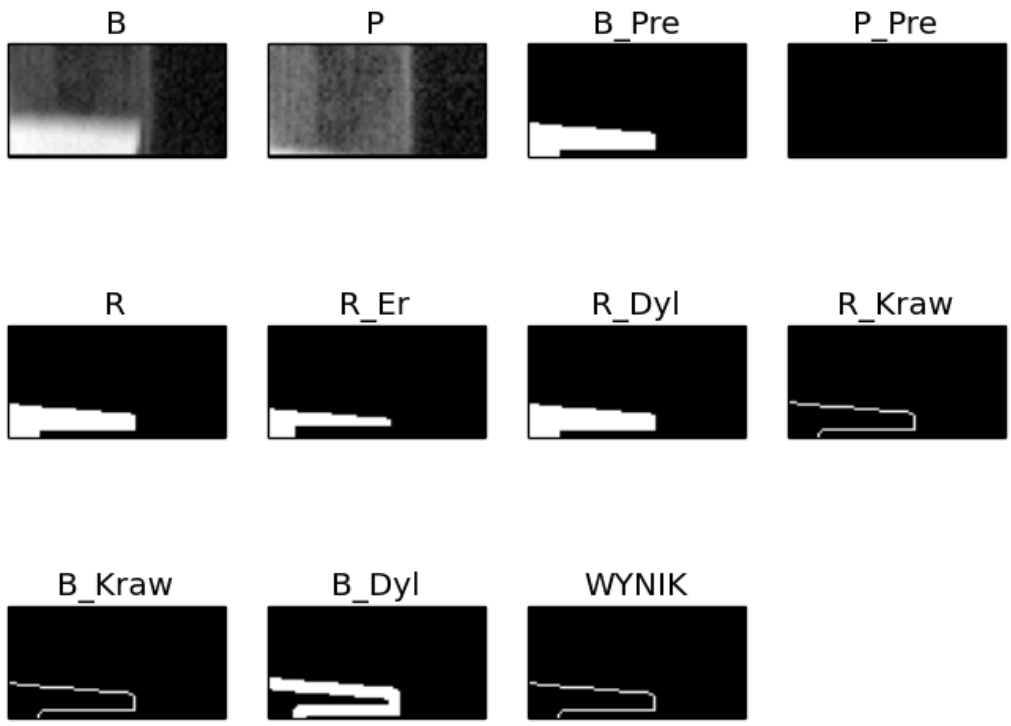
   3. Kierunek jest zaokrąglany do wielokrotności 45°.
3. Usuń piksele, które nie mają maksymalnej wartości – na pewno nie należą do krawędzi.
4. Poddaj obraz progowaniu, aby usunąć za bardzo nachylone krawędzie. Ten proces wykorzystuje histerezę, występują więc dwa progi – górny i dolny.
   1. Jeśli gradient danego piksela jest większy od górnego progu, zaakceptuj go jako część krawędzi.
   2. Jeśli jest poniżej dolnego progu, odrzuć.
   3. Jeśli jest między oboma progami, zaakceptuj, jeśli w jego sąsiedztwie leży inny zaakceptowany piksel.

Jednocześnie zbinaryzowany obraz bieżący również podaje się na ten algorytm, a następnie poddaje się operacji dylatacji. Na posiadanych dwóch wersjach aktualnego obrazu w wykonuje się koniunkcję logiczną. Po tych wszystkich operacjach obraz wynikowy posiada jedynie ślady ruchu. Poniżej zamieszczony został zmodyfikowany wykres z pracy [4].



Rysunek 9: Schemat blokowy stworzonego algorytmu wykrywania ruchu, bazujący na [4].

Wszystkie stadia wyżej opisanego algorytmu znajdują się na rys. 10. Kolejne symbole oznaczają: 'B' – obraz bieżący, 'P' – obraz poprzedni, 'B_Pre' – bieżący po binaryzacji, 'P_Pre', poprzedni po binaryzacji, 'R' – obraz różnicowy, 'R_Er' – obraz różnicowy po erozji, 'R_Dyl' – obraz różnicowy po dylatacji, 'R_Kraw' – obraz różnicowy po wykrywaniu krawędzi, 'B_Kraw' – obraz bieżący po wykrywaniu krawędzi, 'B_Dyl' – obraz bieżący po dylatacji, 'WYNIK' – obraz wynikowy.



Rysunek 10: Przykład działania zaimplementowanego algorytmu.

10. DODATEK B: Dokumentacja techniczna

Dokumentacja oprogramowania

Oprogramowanie służące do wyznaczania prędkości obrotowej wentylatora na podstawie sekwencji zapisanych obrazów zostało napisane w języku Python i testowane z interpreterem 2.7.x. Zarówno kod źródłowy skryptu, jak i powstały na jego podstawie plik wykonywalny są na załączonym do raportu nośniku CD. Poniżej przedstawiono opis oprogramowania:

1. Autor: Łukasz Dudek, data ostatnich zmian: 23.01.2016 r.
2. Przeznaczenie: wykrywanie obecności znacznika na sekwencji obrazów oraz wyznaczanie prędkości obrotowej obiektu.
3. Program działa według algorytmu przedstawionego na rys. 9, a obliczanie prędkości obrotowej opisuje wzór 1. Jest ograniczony do przetwarzania sekwencji obrazów zapisanych

na dysku – nie ma możliwości automatycznego przechwytywania strumienia danych z kamery. Potrzebuje mieć zdefiniowaną ścieżkę do pliku i wartość częstotliwości akwizycji.

4. Argumenty wywołania: ścieżka do katalogu zawierającego obrazy wentylatora. Opcjonalnie: prędkość akwizycji, poziom logowania („DEBUG”, „INFO” lub „WARNING”), wizualizacja.
5. Wartość zwracana: prędkość obrotowa, wyznaczona na podstawie odległości między dwoma obecnościami znacznika mierzona w ilości ramek.
6. Wykorzystywane zmienne globalne: brak.
7. Wywoływane procedury są zamieszczone na poniższej liście:
   a) biblioteka numpy 1.8.x:
      - funkcja ones
      - typ uint8
   b) biblioteka cv2 (OpenCV 2.4.11-7):
      - funkcja imread
      - funkcja cvtColor
      - funkcja blur
      - funkcja threshold
      - funkcja erode
      - funkcja dilatate
      - funkcja subtract
      - funkcja Canny
      - funkcja bitwise_and
      - wartość zmiennej wyliczeniowej IMREAD_GRAYSCALE
      - wartość zmiennej wyliczeniowej THRESH_BINARY
      - wartość zmiennej wyliczeniowej COLOR_BGR2GRAY
   c) biblioteka os (wersja zgodna z wersją interpretera):
      - funkcja listdir
      - funkcja join z modułu path
   d) biblioteka logging (wersja zgodna z wersją interpretera):
      - klasa StreamHandler wraz z metodami
      - klasa FileHandler wraz z metodami
      - klasa Formatter wraz z metodami
      - klasa Logger wraz z metodami

- funkcja getLogger

e) moduł pyplot z biblioteki matplotlib 1.4.2:

- funkcja subplott
- funkcja gray
- funkcja imshow
- funkcja xticks
- funkcja yticks
- funkcja title
- funkcja show

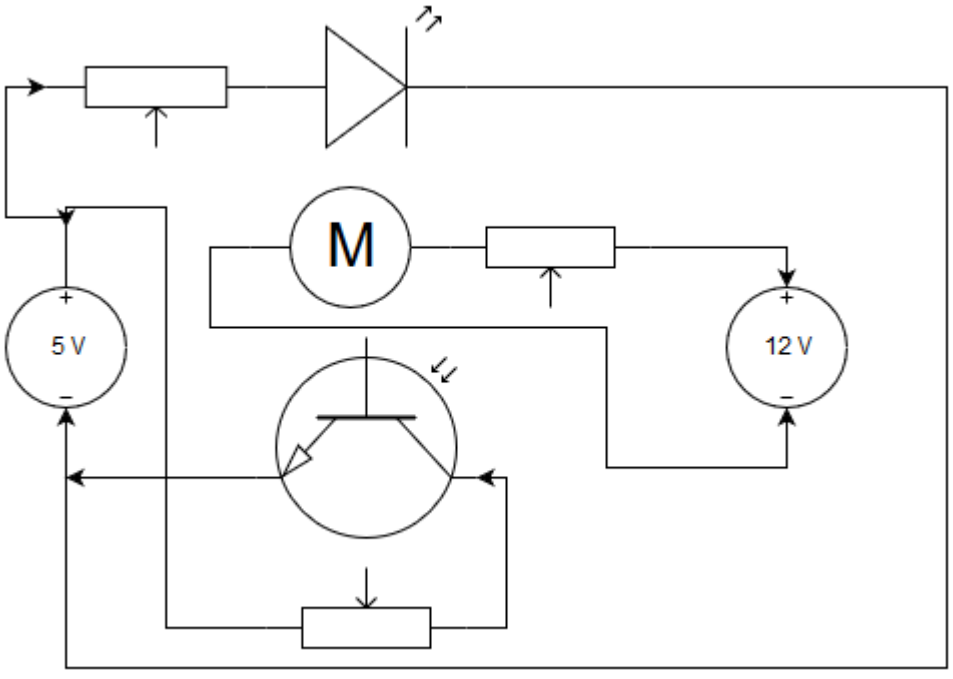
f) biblioteka argparse (wersja zgodna z wersją interpretera):

- klasa Logger wraz z metodami

Dokumentacja sprzętowa

Urządzenie do pomiaru prędkości obrotowej wentylatora

- Autor: Michał Ciszewski, data montażu: 10.12.2015 r.
- Sprzęt wykorzystywany do optycznej detekcji ruchu łopatek wentylatora



Rysunek 11: Schemat połączenia układu detekcji ruchu.

Układ tworzony był z myślą o zasilaniu bezpośrednio z portu USB, dlatego też sugerowanym urządzeniem wykorzystywanym do zasilania jest laboratoryjny zasilacz stabilizowany 5V. Jednak zgodnie ze specyfikacją maksymalne bezpiecznie napięcie zasilania wynosi 6V. Z racji posiadania przez układ odpowiednio dobranych rezystorów podpięcie do

zasilania o napięciu z przedziału (1.5 - 6 [V]) nie powinno wyrządzić nieodwracalnych szkód. Zaleca się jednak nie przekraczanie proponowanych wartości.

Wentylator komputerowy posiada osobny układ zasilania w postaci zasilacza o regulowanym zakresie napięciowym. Zalecane wartości to 6, 9 lub 12 [V]. W inach przypadkach może dojść do spalenia się układu napędzającego. Układ bazujący na transoptorzorze posiada trzy potencjometry, jeden osiowy wieloobrotowy o maksymalnej wartości rezystancji 1k oznaczany dalej jako R1, oraz dwa osiowe jednoobrotowe o rezystancji 10 kΩ - R2 oraz 1kΩ – R3. Potencjometr R3 służy do regulacji prądu przepływającego przez diodę umieszczoną w transoptorze, prąd ten ograniczony jest przez rezystor o wartości 50. Nie pozwala on na spalenie się diody w układzie.

Potencjometry R1 oraz R2 opisywane w specyfikacji transoptora jako “Load resistance” służą do ustawienia czasu odpowiedzi oraz histerezy układu. Im szybsza jest odpowiedź tym mniejsza histereza. Potencjometr R2 służy do wstępnego ustawienia rezystancji, natomiast R1 z racji bardzo wysokiej dokładności do precyzyjnego dojścia do danej żądanej wartości.

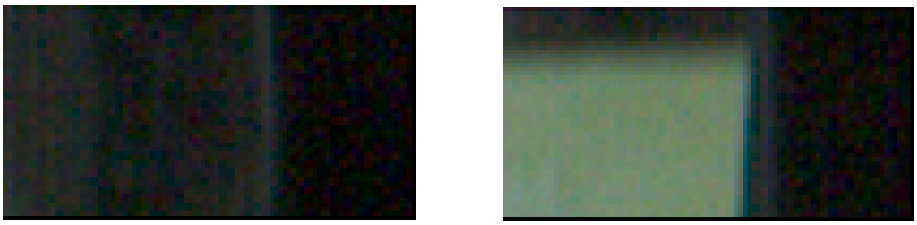
Model wykorzystywanego transoptora to KTIR0911S (dokumentacja w [8]). Najważniejszymi jego parametrami jest szczelina o szerokości 15 mm oraz zakres napięciowy (1 - 6 [V]).

Procedura symulacji i testowania:

Urządzenie do pomiaru prędkości obrotowej obiektu wraz z wentylatorem komputerowym.

1. Na zasilaczu laboratoryjnym ustawić wartość napięcia z przedziału 6- 12 [V]
2. Ustawić ograniczenie prądowe 20 [mA]
3. Podłączyć wentylator do zasilacza laboratoryjnego zwracając szczególną uwagę na kolor przewodów.
4. Za pomocą dołączonego potencjometru ustawić satysfakcjonują prędkość obrotową
5. Podłączyć układ pomiarowy do zasilania **5 V** zwracając uwagę na oznaczenia na płytce.
6. Sprawdzić za pomocą kamery czy świeci się dioda w transoptorze.
7. Korzystając z oscyloskopu podpiętego pod wyjście tranzystora (w transoptorze) używając potencjometrów ustawić odpowiednie czasy reakcji i histerezę układu.
8. Wyjście z tranzystora przepiąć do układu wyzwalającego i używając potencjometru ustawić wartość napięcia referencyjnego na średnią z minimalnego i maksymalnego napięcia z układu pomiarowego (pomiary robić ruszając palcem łopatkę wentylatora)
9. Wyjście z płytki wyzwalającej podłączyć do płytki wyzwalającej kamerę (na stanie laboratorium).

10. Obracając przy pomocy palca wentylator sprawdzić czy czerwona dioda miga.
11. Jeżeli wyniki wszystkich kroków zgadzają się z opisem podpiąć kamerę do płytki wyzwalającej kamerę.
12. Ustawić żądane parametry w Pylonie (pamiętać o włączeniu trybu Trigger). Można również użyć ustawień zamieszczonych na dołączonej płycie CD.
13. Używając potencjometru, zainicjować kręcenie się wentylatora.
14. Wygenerować odpowiednią ilość obrazów (np. 500) o niskiej rozdzielczości zawierających element wskaźnika. Poniżej podano przykładowe dwa obrazki.
15. W przypadku braku możliwości uzyskania pożądanych obrazów istnieją gotowe próbki (patrz podrozdział **Procedura uruchomieniowa i weryfikacji sprzętowej**)



Rys 11. Przykładowe obrazy przesuwającego się znacznika.

Procedura uruchomieniowa i weryfikacji sprzętowej:

Dostarczone w ramach prac nad projektem oprogramowanie działa niezaleznie od połączenia sprzętowego. Dostarczono je w dwóch wersjach: skryptu w języku Python oraz pliku wykonywalnego. Aby uruchomić program “movement_measurement”, należy wykonać następujące kroki:

1. Należy otworzyć okno konsoli. W systemie Windows jest to program “cmd.exe”.
2. Należy przejść do katalogu “src/Python-OpenCV”.
3. Należy wpisać polecenie: “python movement_measurement.py ścieżka_do_obrazów”. To uruchamia program, który przetwarza 500 ramek zebranego obrazu i pokazuje tylko te, na których znajduje znacznik.
4. Alternatywnie, należy przejść do katalogu „EXE/movement_measurement”.
5. Tam wpisać polecenie: „movement_measurement.exe ścieżka_do_obrazów”.

11. DODATEK C: Spis zawartości dołączonego nośnika (płyta CD ROM)

W poszczególnych folderach nośnika, w zależności od rodzaju projektu, znajdują się:

- **SRC – pliki źródłowe dla Arduino, Pythona i wstępnego algorytmu napisanego w C++ przy wykorzystaniu biblioteki OpenCl**
- **EXE – plik wykonywalny z przykładowymi obrazami do detekcji**
- **TEST – 500 obrazów obracającego się wentylatora, ustawienia kamery.**
- **DOC – raport wraz z wykorzystaną literaturą oraz grafiką użytą do tworzenia raportu**

12. DODATEK D: Historia zmian

Tabela 1 Historia zmian.

Autor	Data	Opis zmian	Wersja
Łukasz Dudek	2016-01-20	Utworzono szkielet raportu na podstawie wzorca.	0.0

PR15wsw511			**Karta oceny projektu**
Data	Ocena	Podpis	Uwagi