Komputerowe przetwarzanie wiedzy
Projekt
Monitoring Stereowizyjny z wbudowaną inteligencją

Karol Sydor
Łukasz Tułacz

29 stycznia 2009

1 Wstęp

Cel projektu

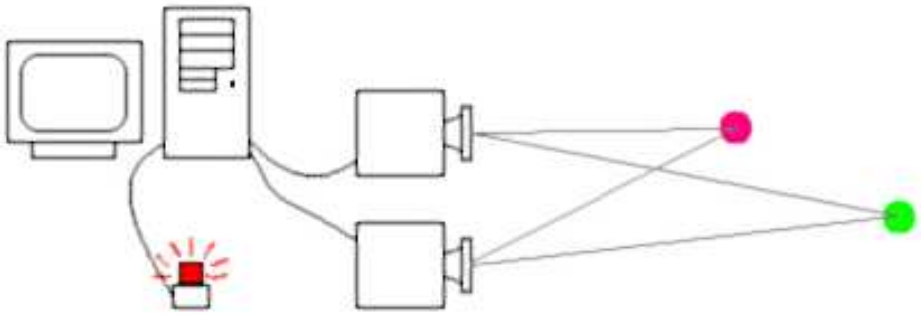
Celem projektu było zaprojektowanie i wykonanie inteligentnego monitoringu stereowizyjnego. System ma przetwarzać obrazy pochodzące z dwóch kamer. Po przeanalizowaniu system określi stopień "zagrożenia".

Założenia

Przyjęto następujące założenia projektowe:

- Głównym elementem przetwarzającym systemu będzie komputer PC
- Jako wejście wykorzystane zostaną dwie identyczne kamery USB
- Jako wyjście zostanie wykorzystana dwukolorowa dioda LED, z płynną zmianą koloru
- Głównym elementem programu będzie oprogramowanie wnioskujące, określające "zagrożenie" na podstawie danych z układu stereowizyjnego
- W celu uproszczenia problemu przetwarzania obrazów, rozpoznawane obiekty będą wcześniej określone
- System ma pracować pod kontrolą systemu Windows, program napisany w środowisku MS VisualStudio 2005 z wykorzystaniem bibliotek OpenCV

2 Realizacja praktyczna



Rysunek 1: Schemat systemu

Uproszczony schemat systemu przedstawiono na rys.1. Obraz z kamer jest przechwytywany przez komputer. Następnie oprogramowanie pracujące w oparciu o model potokowy, przetwarza dwie przechwycone w tym samym momencie ramki obrazu. Efekt wnioskowania jest wizualizowany za pomocą dwukolorowej diody LED. W celu uproszczenia problemu, przyjęto że wykrywanymi obiektami będą dwie piłeczki pingpongowe. Jedna z piłeczek ma kolor zielony - druga różowy. Kolory dobrano tak, aby ułatwić wykrywanie obiektów i uprościć cały problem do znalezienia odpowiedniego koloru na scenie.

Sprzęt

W celu realizacji projektu konieczne było przygotowanie odpowiedniego sprzętu.

Wejście - System kamer

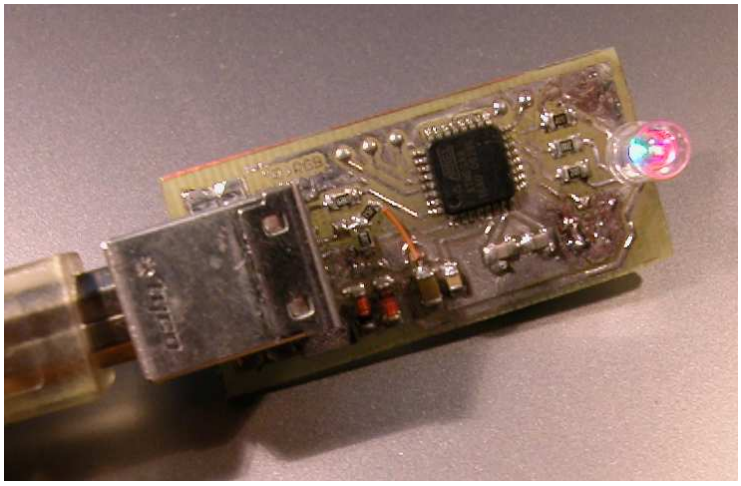
Użyto kamet Logitech QuickCam Connect (rys.2), wyposażonych w przetwornik, o rozdzielczości VGA, oraz interfejs USB. Kamery posiadają możliwość ręcznej regulacji ostrości. Do przechwytywania obrazu wykorzystano funkcje środowiska OpenCV. Obraz przechwytywany jest w tym samym momencie z obu kamer. Rozdzielczość przechwytywanych ramek wynosi 320x240pix. Parametry ekspozycji są dobierane automatycznie przez oprogramowanie wewnętrzne kamer. System unieruchomiono za pomocą drewnianej deseczki (rys.3), z odpowiednio wyciętymi otworami. Znalazło się też miejsce na informację o projekcie (napisaną nieco żartobliwie). Odległość pomiędzy kamerami wynosi 166mm.

Rysunek 2: Kamera Logitech QuickCam Connect

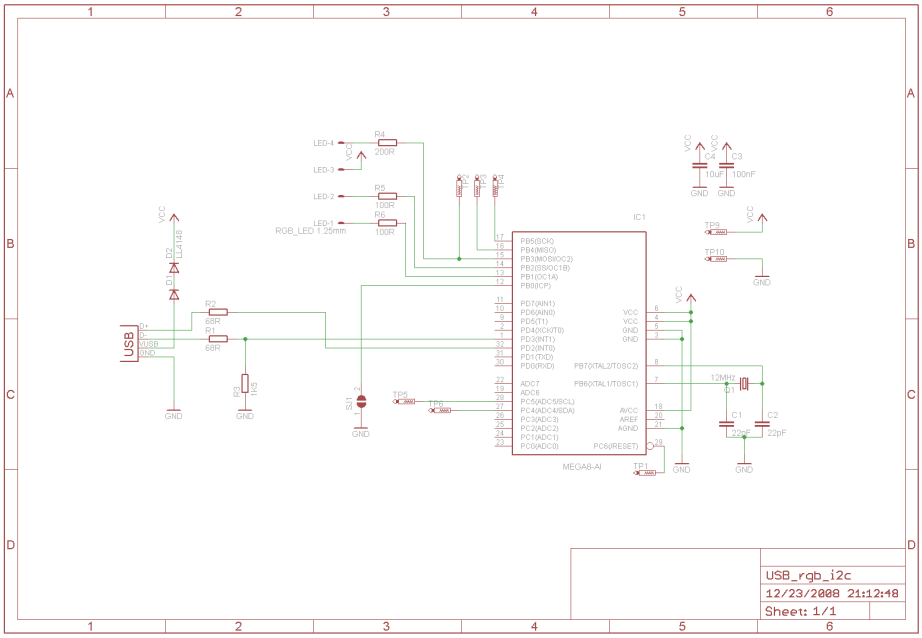
Rysunek 3: Wygląd systemu kamer

Wyjście - Dioda R-G-B

W celu projekcji zagrożenia zbudowano specjalne urządzenie. Jest to dioda RGB podłączana za pomocą interfejsu USB do komputera PC (rys.4). W celu projekcji zagrożenia wykorzystywane są tylko kolory czerwony i zielony. Kolor niebieski jest wykorzystywany wyłącznie do sygnalizacji poprawnego podłączenia. Schemat interfejsu przedstawiono na rys.5. Sercem urządzenia jest 8 bitowy mikrokontroler AVR - Atmega8, taktowany kwarcem 12MHz. Co



Rysunek 4: Dioda LED do projekcji zagrożenia



Rysunek 5: Schemat interfejsu diody LED

prawda nie posiada on sprzętowego interfejsu USB, jednak jest on zrealizowany na drodze programowej. Urządzenie jest zasilane z portu USB. W celu obniżenia napięcia do wartości ok. 3.6V (poprawne napięcie dla interfejsu USB), zastosowano dwie diody prostownicze. Dioda LED sterowana jest za pomocą 3 kanałów sprzętowego PWM. Dzięki temu możliwa jest realizacja 24-bitowego koloru świecenia, za pomocą 3 składowych barw. Oprogramowanie zostało

oparte o gotowy sterownik programowy CDC-IO firmy Recursion Co. Ltd.[1] Oprogramowanie zmodyfikowano tak, aby wykorzystywało 3 kanały sprzętowego PWM mikrokontrolera, oraz rozpoznawało inny układ komend. Komunikacja ze sterownikiem odbywa się na zasadzie wirtualnego portu COM. Po podłączeniu i zainstalowaniu sterowników, wystarczy nawiązać komunikację za pomocą terminala, na odpowiednim porcie. Parametry komunikacji to baud:9600, 1 bit stopu i startu, bez bitów parzystości. Interfejs rozpoznaje komendy w formacie ASCII. Przykładowo:
`00 B 80 R ff G`$[\texttt{CR} + \texttt{LF}]$ - komenda spowoduje wygenerowanie koloru ze składowych: 0 niebieskiego, 0x80 = 128 czerwonego i 0xff = 255 zielonego. Komendy należy zakończyć znakiem Enter (CR+LF).

Oprogramowanie

Oprogramowanie przygotowano w środowisku VisualStudio2005[2] z wykorzystaniem bibliotek OpenCV[3].

Rysunek 6: Schemat oprogramowania

Diagram przedstawiający przetwarzanie danych został pokazany na rys.6 Realizację algorytmu można podzielić na 6 etapów.

Pobieranie obrazu z kamer

W pierwszym podejściu, w celu przechwycenia obrazu wykorzystano funkcje
`cvGrabFrame(cvCaptureFromCAM(0));`
która przechwytuje obraz z pierwszej dostępnej kamery (kolejne wywołanie z indeksem 1 powoduje pobranie obrazu z drugiej kamery). Następnie za pomocą funkcji
$*$`IplImage = cvRetrieveFrame(capture)`
następowało pobranie przechwyconego obrazu do postaci IplImage, która jest podstawową zmienną zawierającą obraz w środowisku OpenCV. Środowisko OpenCV dba o to żeby obraz był przechwycony w tym samym momencie z obydwu kamer. Jednak w trakcie pracy okazało się że ten sposób przechwytywania ramek nie daje zadowalających efektów. Co prawda uzyskano rozdzielczość 640x480pix, ale prędkość pobierania klatek oscylowała w okolicy 1 klatki/

[1] Dostępne na licencji GNU GPL 2, treść licencji oraz zmodyfikowane na potrzeby projektu oprogramowanie, dostarczono wraz z oprogramowanie projektu
[2] Dostępny za darmo dla studentów, w ramach programu MSDNAA
[3] Rozpowszechniany na licencji GPL

sekundę. Na dodatek pojawiało się znaczne opóźnienie pomiędzy pobieranym obrazem, a rzeczywistością. W drugim podejściu zmieniono sposób przechwytywania ramek. Wykorzystano funkcję
`stereocallback(image1, image2)`
Jest to funkcja typu callback, wywoływana po poprawnym przechwyceniu ramek. Co prawda rozdzielczość przechwytywanego obrazu wynosi tylko 320x240pix, ale za to udało się wyeliminować niekorzystne opóźnienie i znacząco zwiększyć prędkość. Udało się uzyskać zadowalającą prędkość przetwarzania na poziomie 6-10 ramek/sekundę. Zaobserwowano ciekawą własność - żeby kamery pracowały poprawnie, muszą być podłączone do głównego koncentratora USB. Mogą być podłączone poprzez HUB-y, jednak muszą być podłączone do 2 niezależnych kanałów głównego koncentratora.

Przetwarzanie wstępne

Obraz pobrany z kamery jest łatwy do przetworzenia dla ludzkiego oka, ale pełen zbędnych informacji dla komputera. W celu normalizacji obrazu, należy przetworzyć obraz. Wykorzystano rozmywanie gaussowskie:
`cvSmooth(IplImage, IplImage, CV_BLUR, 3);`
Następnie, w celu ekstrakcji poszukiwanych kolorów wykonywane są operacje na macierzach saturacji i tonu barwy. Dla piłeczki różowej dokonane jest przesunięcie barwy w kanale czerwonym o +10, następnie dwukrotne wzmocnienie nasycenia. Dla piłeczki zielonej dokonane jest przesunięcie barwy w kanale zielonym o -80, następnie dwukrotne wzmocnienie nasycenia. W ten sposób przygotowane obrazy, są łatwe do dalszej obróbki.

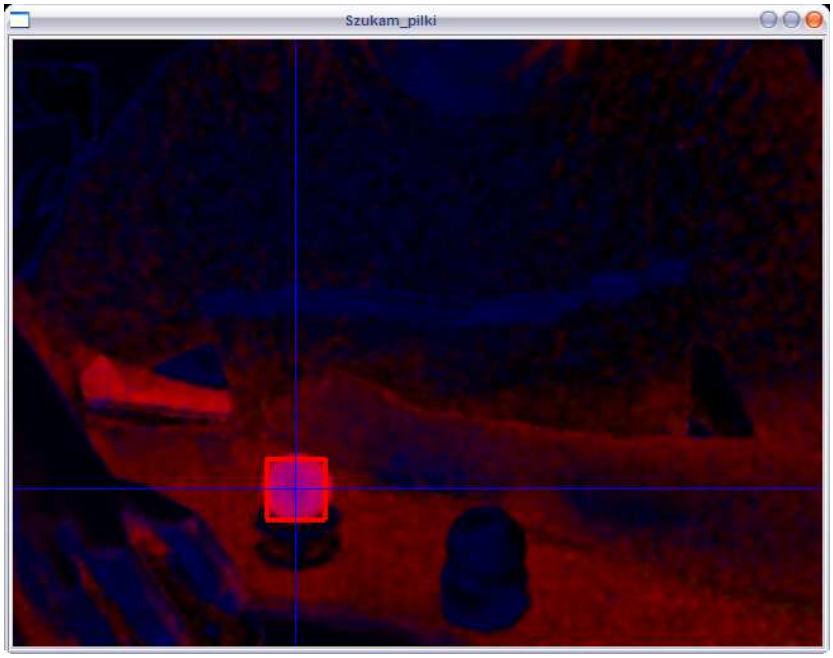
Wyszukiwanie obiektów

Poszczególne piksele obrazu są poddawane operacji progowania. Jest to znacznie uproszczone dzięki odpowiedniej budowie zmiennej IplImage. Do poszczególnych pikseli odwołujemy się jak do tablicy jednowymiarowej. Arbitralnie dobrano wartości progowe. Wykrywamy różową piłeczkę jako złożenie dwóch kolorów - czerwonego i niebieskiego. Zielona piłeczka wykrywana jest jako wyłącznie zielony obszar. Następnie odnajdywane są brzegi piłeczki, oraz wyliczany jest środek. Cały proces pokazano na rys.7, 8 i 9.

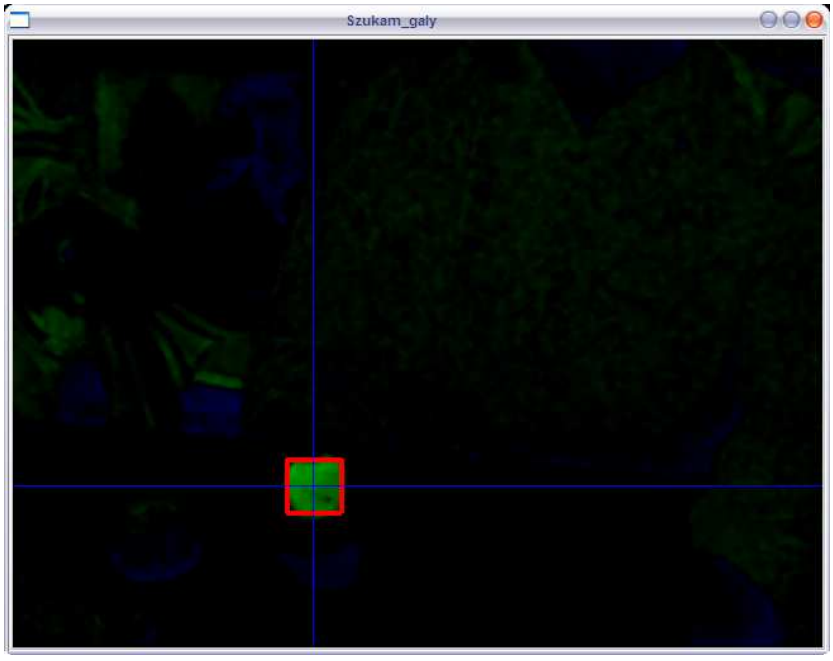
W trakcie prowadzenia badań zaobserwowano konieczność korekcji ustawień kamery, w celu adaptacji do konkretnego oświetlenia. Na szczęście sterowniki dostarczone przez producenta umożliwiają kontrolowanie parametrów kamery w bardzo szerokim zakresie, łącznie z całkowitym wyłączeniem automatyki.

Kalkulacja odległości

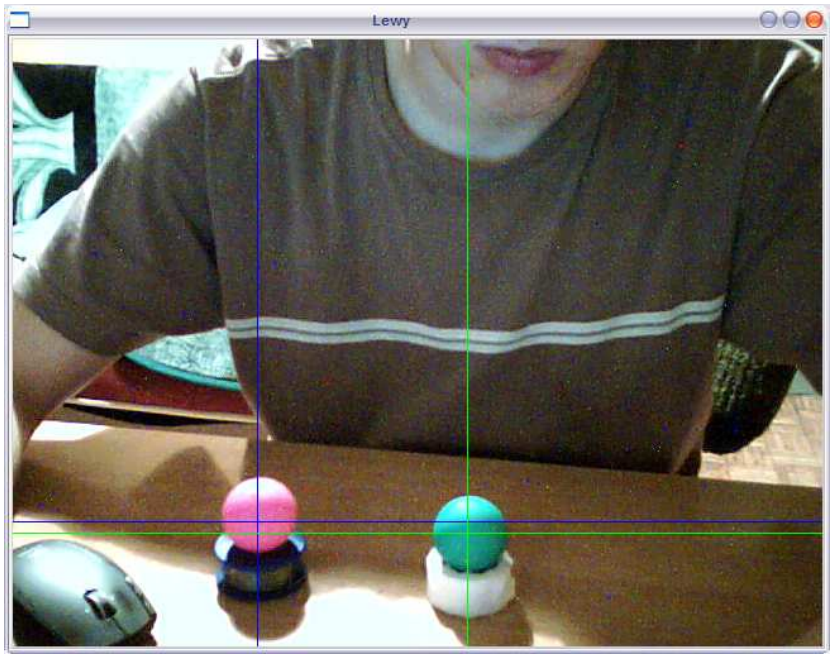
Kalkulacja odległości odbywa się w oparciu o zagadnienie stereowidzenia. Dzięki przyjętej konwencji wyszukiwania obiektów nie musimy stosować skomplikowanych algorytmów dopasowania. Zabiegi opisane powyżej ustalają, konieczne do określenia odległości, współrzędne piłek na obu obrazach. W takim przypadku zadanie pomiaru odległości spełnia triangulacja. Przed omówieniem triangulacji należy jednak przedstawić parametry oraz zależności związane z rejestrowaniem obrazu przez kamery video. Na rys.10 przedstawiono model rzeczywisty kamery video gdzie przyjęto następujące oznaczenia: k - długość



Rysunek 7: Widok okna z obrazem obrobionym w celu znalezienia różowej piłeczki

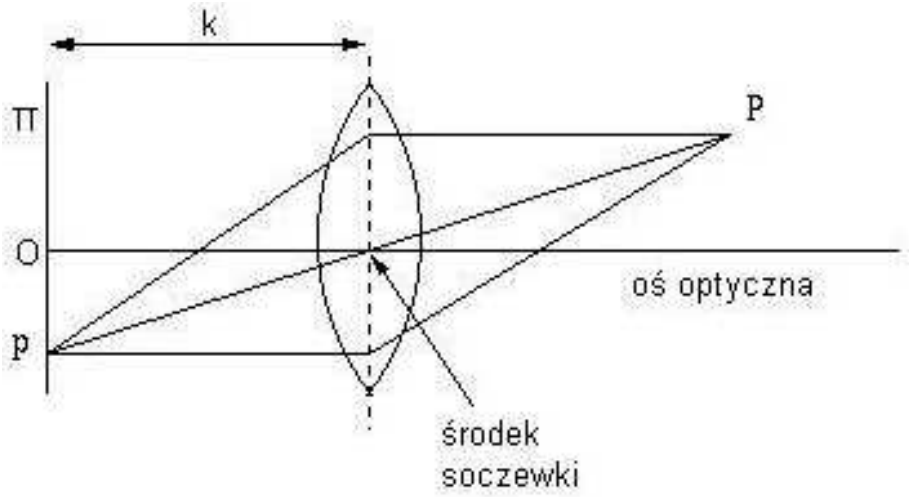


Rysunek 8: Widok okna z obrazem obrobionym w celu znalezienia różowej piłeczki



Rysunek 9: Efekt procesu wyszukiwania piłeczek

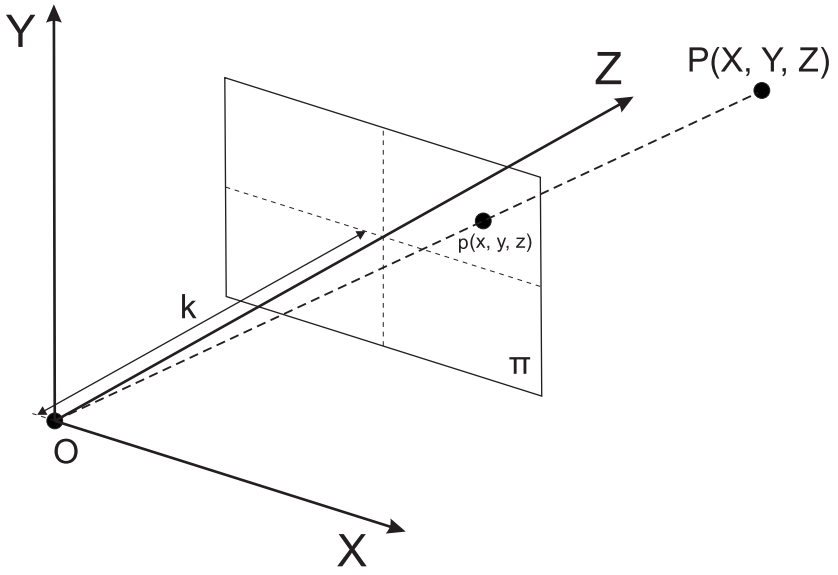
ogniskowej, π - matryca kamery, O - środek matrycy,P - punkt w przestrzeni, p - reprezentacja punktu P na matrycy kamery.



Rysunek 10: Model rzeczywisty kamery

Model rzeczywisty kamery niezbyt dobrze obrazuje zależności oraz parametry które można wykorzystać do pomiaru odległości. Dlatego obraz rejestrowany przez pojedynczą kamerę przedstawia się jako rzut perspektywiczny sceny. Na rys.11 pokazano model przekształcenia perspektywicznego w którym k oznacza

długość ogniskowej, stanowiącą odległość środka rzutowania O od płaszczyzny obrazu π.Prosta przechodząca przez punkt O i środek matrycy nazywa się osią optyczną.Punkt $p(x, y, z)$ jest obrazem punktu $P(X, Y, Z)$ na płaszczyźnie obrazu.



Rysunek 11: Model perspektywiczny kamery

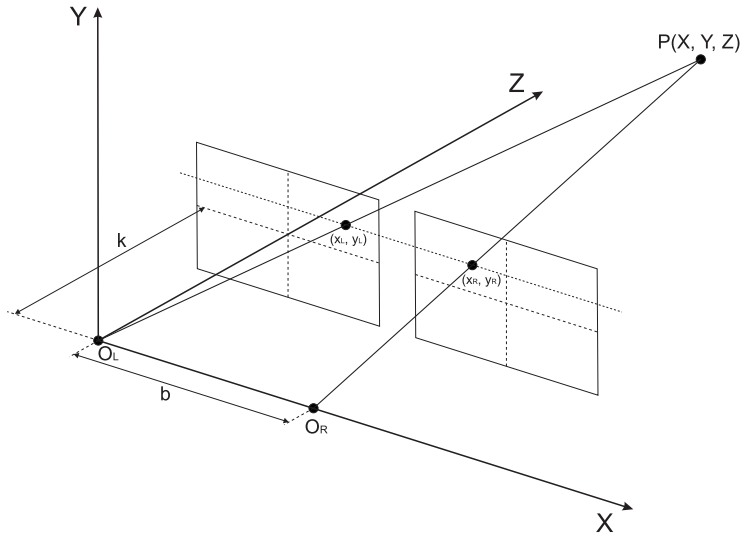
Dla przyjętego, zgodnie z rys.11, układu odniesienia kamery można wyznaczyć następujące zależności:

$$x = k\frac{X}{Z},\ y = k\frac{Y}{Z},\ z = k$$

Aby wyznaczyć zależności umożliwiające obliczenie odległości między punktami w przestrzeni wykorzystując dwie kamery, należy połączyć dwa modele perspektywiczne pojedynczych kamer. Następnie powiązać z sobą współrzędne punktów w przestrzeni, ze współrzędnymi reprezentacji tych punktów na obrazach kamer. Powstały w ten sposób model systemu stereo (rys.12) ma trzy układy odniesienia:

- Globalny układ odniesienia(X, Y, Z), którego początek znajduje się w środku rzutowania lewej kamery(jest to ustandaryzowane), a jednostką są metry lub pochodne,
- Układ odniesienia kamery(x, y, z), którego początek znajduje się w środku rzutowania kamery, a jednostką są metry lub pochodne,
- Układ odniesienia obrazu(x, y), którego początek znajduje się w środku płaszczyzny obrazu, a jednostką są piksele.

W celu powiązania ze sobą różnych układów odniesienia należy znać parametry wewnętrzne oraz zewnętrzne zarówno pojedynczych kamer jak i złożonego z nich sytemu stereo. W przypadku pojedynczej kamery parametry zewnętrzne



Rysunek 12: Model systemu stereo

definiują położenie i orientację układu odniesienia kamery w stosunku do globalnego układu odniesienia. Podstawowe parametry tego przekształcenia to:

- T wektor translacji opisujący względne położenie środków dwóch układów odniesienia,
- R macierz obrotu o wymiarze $3x3$, macierz ortogonalna powodująca nałożenie odpowiadających sobie osi układów.
- k długość ogniskowej

Relacja pomiędzy współrzędnymi punktu P w globalnym układzie współrzędnych P_G oraz układzie współrzędnych kamery P_K jest następująca:

$$P_K = R(P_G - T)$$

W przypadku systemu stereo parametry zewnętrzne opisują względną pozycję i orientację dwóch kamer wykorzystanych w systemie. Podstawowe parametry tego przekształcenia to:

- b odległość między środkami rzutowania(rozbieżność),
- R macierz rotacji pomiędzy układami odniesienia kamer(Kiedy osie optyczne kamer w układzie stereo są względem siebie równoległe macierz R przyjmuje wartość 0.),
- k wspólna długość ogniskowej

Parametry wewnętrzne w obu przypadkach charakteryzują optyczne, geometryczne i cyfrowe cechy kamery. Do parametrów wewnętrznych należą:

- (o_x, o_y) współrzędne środka obrazu [pix]
- (s_x, s_y) rozmiar piksela matrycy kamery [mm]

Dzięki nim możemy powiązać współrzędne punktu na płaszczyźnie obrazu z współrzędnymi w układzie odniesienia kamery. Relacja pomiędzy współrzędnymi punktu obrazu p w układzie odniesienia obrazu (x_{im}, y_{im}) i w układzie odniesienia kamery (x, y) wygląda następująco:

$$x = (x_{im} - o_x)s_x$$
$$y = (y_{im} - o_y)s_y$$

Znając parametry kamer oraz układu stereo możemy przystąpić do pomiaru odległości. Zadaniem pomiaru odległości jest odnalezienie położenia w przestrzeni punktu P na podstawie jego rzutów (x_L, y_L) i (x_R, y_R) (rys.12). Kiedy osie optyczne kamer są równoległe system przedstawiony na rys.12 można opisać zależnością:

$$\frac{X}{x_L} = \frac{X-b}{x_L-d} = \frac{Y}{y_L} = \frac{Z}{k}$$

gdzie:

$$d = x_L - x_R$$

Po przekształceniach otrzymujemy wzory na współrzędne punktu P w przestrzeni trójwymiarowej:

$$X = \frac{x_R}{d}b,\ Y = \frac{y_L}{d}b,\ Z = \frac{k}{d}b$$

Bywa, że parametry wewnętrzne lub zewnętrzne nie są znane. W takiej sytuacji należy je wyznaczyć. Proces wyznaczania parametrów wewnętrznych i zewnętrznych nazywa się kalibracją. Producenci kamer internetowych bardzo rzadko podają potrzebne parametry. Metody wyznaczania są bardzo zróżnicowane. Poniżej zostaną przedstawione tylko wybrane metody wyznaczania parametrów które są niezbędne do określenia odległości. Sposób wyznaczania długości ogniskowej przedstawiono na rys.13. W tym celu ustawiamy dwa przedmioty o znanych szerokościach l w różnych znanych odległościach od obiektywu kamery g_1, g_2. Otrzymamy w ten sposób reprezentacje tych przedmiotów na obrazie o szerokościach x_1 i x_2 we współrzędnych obrazu [pix]. Znając je obliczamy długość ogniskowej k, wprost ze wzorów:

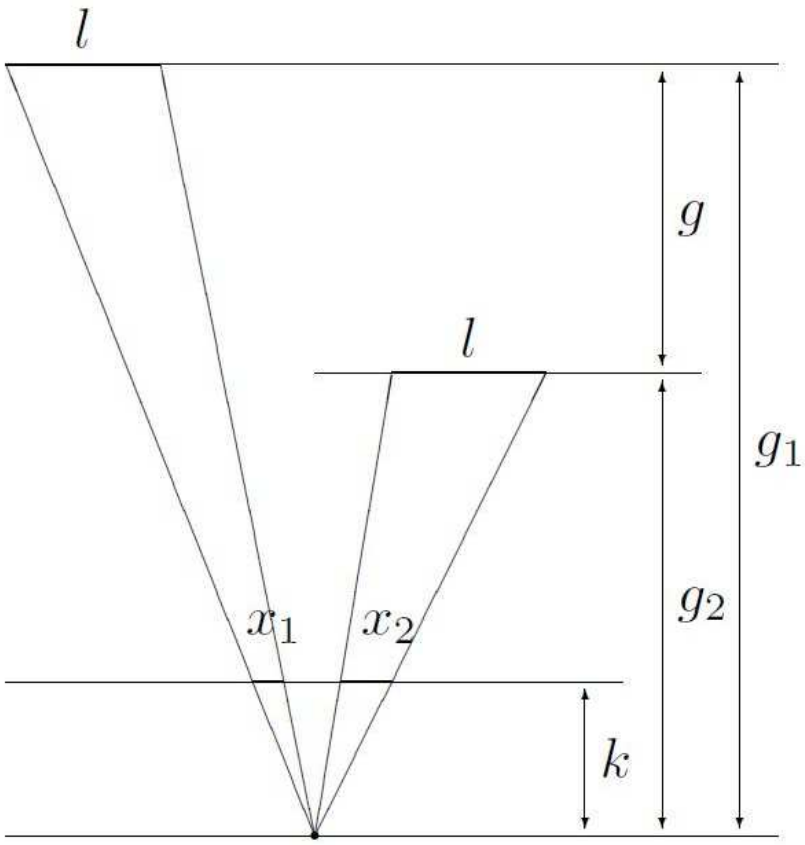
$$g = g_1 - g_2 = kl(\frac{1}{x_1} - \frac{1}{x_2})$$
$$k = \frac{gx_1x_2}{lx_2-x_1}$$

Innym parametrem często wymagającym kalibracji jest rozmiar piksela matrycy kamery. Realizuje się to poprzez dokonanie pomiaru odległości wybranych punktów sceny przy znanych odległościach rzeczywistych. Ustawiamy mały przedmiot(n.p. kulkę) nieruchomo względem kamery na linii osi optycznej, po czym mierzymy odległość kulki od kamery. Jest to wartość współrzędnej Z kulki w przestrzeni. Przekształcając podany wcześniej wzór na wsp. Z przy założeniach że $o_y = o_x$ i $s_y = s_x$ wzór na wielkość piksela matrycy kamery wygląda następująco:

$$s = \frac{kb}{Z(x_L-x_R)}$$

Wyliczone parametry dla wykorzystanych kamer:

- Długość ogniskowej - 4mm.



Rysunek 13: Wyznaczanie długości ogniskowej

- Rozmiar piksela matrycy - 0.0065 mm przy rozdzielczości 640x480 (dwukrotnie większy przy 320x240)
- Wykorzystana rozdzielczość kamery - 320x240pix
- Rodzaj matrycy kamery - CMOS.

Wnioskowanie

W celu przeprowadzenia wnioskowania zaproponowano algorytm oparty o logikę rozmytą. Dane pochodzące z algorytmów stereowizji - odległość wejściowa i prędkość wejściowa - są przeliczane na postać rozmytą. Proces zilustrowano na diagramach (rys.15 i 14). Dane wejściowe są poddawane procesowi przetwarzania. Ilustruje to czerwona linia na diagramach. Proces przeliczania można opisać wzorem:

$$\text{Prędk.wej.} < \text{wolno} -> \text{Prędkość} = 0$$
$$\text{Prędk.wej.} > \text{szybko} -> \text{Prędkość} = 1$$

Wewnątrz przedziału (wolno,szybko) "Prędkość" przyjmuje wartości pośrednie.

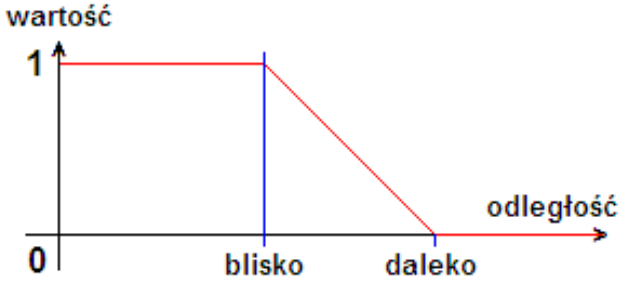
Dla przypadku wyliczania odległości, postępujemy analogicznie, jednak funkcja jest inna - poniżej progu blisko funkcja przyjmuje wartość 1, a powyżej daleko

- wartość 0. Progi: szybko, wolno, blisko i daleko dobrano arbitralnie. Po etapie rozmywania, prowadzona jest operacja wyliczania wartości wyjściowej. Jest to dokonywane za pomocą prostego wzoru.

$$y = 1 * odleglosc + 0.5 * predkosc$$

Tak wyliczona wartość funkcji y jest zależna od odległości - jeśli piłeczka jest zbyt blisko, wyświetlany jest stan zagrożenia. Jednak w pewnym stopniu ta funkcja jest zależna też od prędkości - jeśli piłeczki zbliżają się do siebie wystarczająco szybko, wtedy również odczytywane jest to jako zagrożenie.

Rysunek 14: Rozmywanie zmiennej prędkości



Rysunek 15: Rozmywanie zmiennej odległości

Projekcja zagrożenia

Funkcja odpowiedzialna za projekcję zagrożenia przetwarza wartość zmiennej y otrzymanej w wyniku wnioskowania. Wartość ta jest odpowiednio wzmacniana

$$y = y * 255$$

Wartość 255 odpowiada kolorowi maksymalnie czerwonemu, 0 odpowiada kolorowi całkowicie zielonemu. Zależność dana jest wzorem:

$$R = y, G = 255 - y$$

W przypadku kiedy brakuje danych wejściowych - nie zlokalizowano piłeczek - dioda jest wygaszana.

3 Pliki projektu

Wymagania i konfiguracja oprogramowania projektu

W celu uruchomienia oprogramowania projektu, potrzebne są:

- Visual Studio 2005 (C++)
- Zainstalowane biblioteki OpenCV
- Dwie zainstalowane kamery, zgodne z OpenCV

W trakcie prac nad projektem korzystaliśmy z bibliotek OpenCV w wersji 1.1pre1. Instalacja przebiega w standardowy dla systemu Windows sposób. Instrukcja instalacji znajduje się na stronie OpenCV (link w sekcji Literatura i odnośniki).

W celu uruchomienia i edycji projektu, należy otworzyć plik: "Sztuczna_inteligencja.sln" za pomocą środowiska Visual Studio. Następnie należy postępować zgodnie z procedurą dołączenia plików opencv do projektu. Zostało to dokładnie opisane (od kroku 7) na stronie: http://opencv.blogspot.com/2006/04/getting-started-with-opencv-in-visual.html. W przypadku błędów kompilacji spowodowanych przekształceniami LPCWSTR, należy przejść do ustawień projektu: Project/Properties i w zakładce Configuration Properties/General, ustawić opcję "Character Set" na "Use Multi-Byte Character Set".

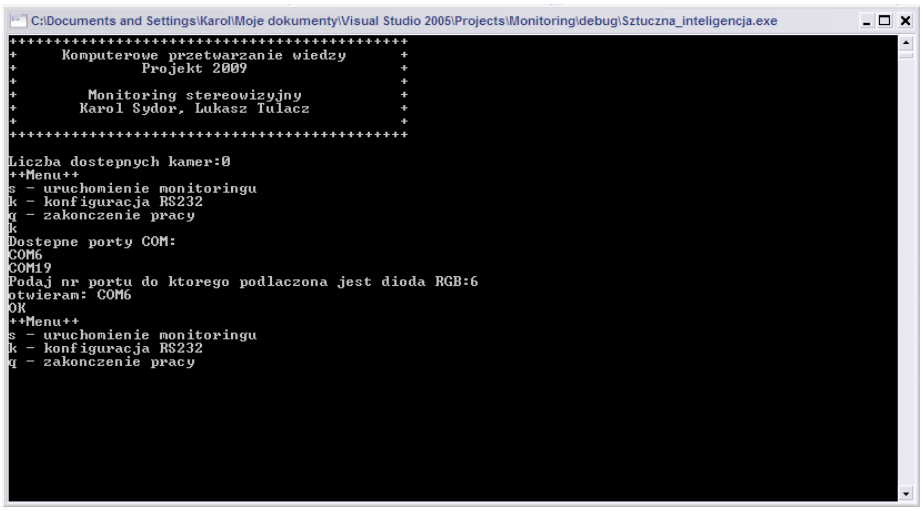
Żeby uruchomić projekt można też skorzystać z gotowego pliku wykonywalnego - "Projekt.exe". Znajduje się on w głównym folderze projektu.

Obsługa programu projektu

Wygląd okna głównego programu pokazano na rys.16. Po uruchomieniu, wyświetla się dostępna liczba kamer (powinno być 2) oraz proste menu tekstowe. W celu wyjścia z programu należy wybrać klawisz "Q" (program reaguje na małe i wielkie litery) i potwierdzić ENTER. W celu uruchomienia systemu należy najpierw wybrać "K" (konfiguracja RS232). Wyświetlona zostanie lista dostępnych w systemie portów RS232. Należy wybrać numer odpowiedni podłączonemu interfejsowi diody LED (można to sprawdzić korzystając z menedżera urządzeń). W przypadku poprawnej konfiguracji system wyświetli odpowiedni komunikat i wróci do menu. Podłączona dioda zabłyśnie niebieskim światłem. Po tej operacji można uruchomić system "S". Jeżeli podłączone są dwie kamery i poprawnie skonfigurowano interfejs, uruchomione zostanie przetwarzanie. Wyświetli się okno z podglądem z lewej kamery. W celu przerwania symulacji należy wcisnąć klawisz "Q" i potwierdzić ENTER, przy aktywnym oknie głównym programu.

Projekt diody RGB

Wszystkie pliki potrzebne do odtworzenia interfejsu diody USB zostały dostarczone na płycie projektu. Płytkę drukowaną można wykonać samodzielnie, projekt płytki jest w formacie programu Eagle 5.4. Skompilowane oprogramowanie dla mikrokontrolera znajduje się w pliku "firmware.hex" w katalogu "Oprogramowanie dioda RGB". Kompletny projekt oprogramowania



Rysunek 16: Przykładowy wygląd okna programu

w formacie programu AVR-Studio znajduje się w tym samym folderze. Instrukcja konfiguracji fusebit-ów mikrokontrolera dostępna jest na stronie: http://www.recursion.jp/avrcdc/. Po wgraniu oprogramowania i ustawieniu fusebit-ów, interfejs jest gotowy do pracy. Instalacja w systemie Windows wymaga sterownika, znajduje się on w folderze "Oprogramowanie dioda RGB/inf" w

4 Podsumowanie

Rozbieżności z założeniami

Na początku planowano zrealizować projekt w nieco innym kształcie. Miał to być system monitoringu, który będzie analizował położenie określonego przedmiotu, w zależności od wybranych "stref". Jednak z czasem przyjęto uproszczenia. Z określonych obiektów powstały konkretne piłeczki, o jednolitym kolorze. Ze stref zrezygnowano, starając się określić jedynie zależności pomiędzy piłeczkami. Rozbieżności nie biegną jedynie w kierunku uproszczeń - rozbudowano nieco system wnioskujący, który w początkowych założeniach był banalny (oparty jedynie na położeniu). Pomimo że w obecnej postaci jest bardzo prosty algorytmicznie, spełnia doskonale swoje zadanie. Nie reaguje jedynie na położenie, ale stara się też odgadnąć "tendencję" piłeczek, reagując na prędkość. Odpowiedni dobór współczynników sprawił że wnioskowanie działa dobrze w praktyce.

Wnioski

Końcowy efekt spełnił nasze oczekiwania. Pomimo że system działa dość topornie przy oświetleniu innym niż idealne (to w którym był opracowywany), daje się zauważyć poprawne jego reakcje. Algorytmy stereowizyjne dość dobrze określają odległość. Największy problem sprawiały elementy, które teoretycznie powinny pójść gładko - uruchomienie kamer w układzie stereowizji, oraz samo

przetwarzanie obrazów. Odpowiednia praca kamer dała się zrealizować dopiero przy drugim podejściu. Kiedy reszta programu była już ukończona, w fazie eksperymentów udało się uruchomić stereocallback. Powód wcześniejszych problemów był dość prozaiczny - wystarczyło wyłączyć podgląd na drugiej kamerze. Gotowy program przerobiono pod nowe rozwiązanie. Zaowocowało to znacznym wzrostem szybkości przetwarzania. Obecnie reakcja systemu jest realizowana w czasie zbliżonym do rzeczywistego - opóźnienie jest niewielkie. Wcześniej było kilkusekundowe, a szybsze ruchy piłeczek były w ogóle niezauważalne dla systemu.

Literatura i odnośniki

- Materiały do wykładu "Systemy Wizyjne". Marek Wnuk, Wrocław
- "Przetwarzanie informacji wizyjnej w komputerowym systemie z mobilną głowicą stereowizyjną" Przemysław Kowalski, Krzysztof Skabek - Instytut Informatyki Teoretycznej i Stosowanej PAN
- "Learning OpenCV: Computer Vision with the OpenCV Library", Gary Bradski, Adrian Kaehler
- Archiwum listy mailingowej osdir: http://osdir.com/ml/lib.opencv/
- Interfejs USB: http://www.recursion.jp/avrcdc/
- Informacje i download OpenCV: http://opencv.willowgarage.com/wiki/
- Opis konfiguracji OpenCV i Visual Studio 2005: http://opencv.blogspot.com/2006/04/getting-started-with-opencv-in-visual.html