Raport z projektu indywidualnej organizacji studiów

Wojciech Ptasiński Maciej Ziaja

12 sierpnia 2020

Streszczenie

Abstract

Spis treści

0.1 Wstęp

0.2 Środowisko prototypowania robotów ROS

0.3 Filtr Kalmana i system AHRS

System AHRS (ang. *attitude and heading reference system*) pozwala na wyznaczenie orientacji obiektu w przestrzeni. Systemu tego typu znajdują szerokie zastosowanie w pojazdach autonomicznych.

System AHRS można podzielić na dwie główne części: system czujników oraz system predykcji orientacji w przestrzeni. Fragment systemu odpowiedzialny za działanie czujników stawia wyzwania związane z obsługą sprzętu oraz jego kalibracją. Algorytm predykcji pozwala na fuzję danych z zespołu czujników w celu określenia orientacji obiektu w przestrzeni.

Wynikiem działania systemu AHRS jest wektor trzech wartości reprezentujących obrót obiektu wokół osi trójwymiarowego układu współrzędnych. Kąty obrotu noszą nazwę *kątów Eulera*, ich ideowym odpowiednikiem w terminologii lotniczej są kąty *roll, pitch, yaw*.

0.3.1 Czujniki systemu AHRS i implementacja sprzętowa

W celu określenia orientacji obiektu w przestrzeni konieczny jest zestaw czujników. Do wyznaczenia pełnej orientacji w przestrzeni trójwymiarowej stosuje się zazwyczaj zestaw: akcelerometru, żyroskopu oraz magnetometru. Akcelerometr i żyroskop tworzą razem zestaw nazywany IMU (ang *inertial measurement unit*). System IMU pozwala na wyznaczenie kątów odchylenia obiektu od pionu (*roll* oraz *pitch*), co w wielu zastosowaniach jest wystarczające. Aby uzyskać pełne współrzędne orientacji obiektu w przestrzeni należy użyć także magnetometru, który pozwala określić kąt obrotu wokół osi pionowej (*yaw*).

W projekcie użyto układu *MinIMU-9 v5* firmy *Pololu* o dziewięciu stopniach swobody (trzy stopnie swobody na każdy z czujników). *MinIMU* zawiera zestaw: układ IMU *LSM6DS33* oraz magnetometr *LIS3MDL*. Komunikacja z czujnikami odbywa się poprzez magistralę *I2C*, przy czym każdy z trzech czujników ma osobny adres i wymaga oddzielnej obsługi komunikacji. Czujniki wchodzące w skład systemu AHRS wymagają kalibracji, co jest szczególnie ważne przypadku magnetometru.

Żyroskop

Żyroskop to przyrząd pozwalający określić prędkość obiektu kątową w trzech osiach. W spoczynku żyroskop powinien dawać zerowe odczyty ze wszystkich osi. Jest to podstawowy czujnik pozwalający określić orientację obiektu w przestrzeni. Wyznaczenie kątów obrotu wymaga jedynie całkowania odczytu z żyroskopu (ponieważ prędkość kątowa to pochodna kąta obrotu). Niestety sam żyroskop nie jest wystarczający aby precyzyjnie orientować obiekt w przestrzeni, ze względu na zjawisko dryftu (ang. *gyroscope drift*). Żyroskop jest mało podatny na szumy pomiarowe o małej częstotliwości, natomiast jest wrażliwy na pojawianie się stałych, ale niewielkich błędów w odczytach. Ze względu na konieczność całkowania danych z żyroskopu te małe błędy są kumulowane, w wyniku czego wraz z upływem czasu, orientacja na podstawie działania żyroskopu staje się coraz mniej wiarygodna. Dlatego w celu precyzyjnego ustalania orientacji w przestrzeni konieczne jest użycie żyroskopu w parze z akcelerometrem oraz zastosowanie algorytmu fuzji danych.

Akcelerometr

Akcelerometr to czujnik pozwalający mierzyć przyspieszenie kątowe obiektu. W stanie spoczynku osie poziomie akcelerometru powinny wskazywać zerowe odczyty, natomiast oś pionowa powinna wskazywać wartość przyspieszenia ziemskiego.

Akcelerometr jest wolny od zjawiska dryftu, natomiast jest wrażliwy na szumy pomiarowe o dużych częstotliwościach. Przeciwna natura niedokładności akcelerometru oraz żyroskopu sprawia, że najlepiej pracują one w parze, z użyciem algorytmu fuzji danych.

W czujnikach IMU często dokonuje się kalibracji błędu zera (addytywnego). Należy wykonać to w trakcie spoczynku systemu na płaskiej powierzchni. Oś pionowa akcelerometru nie powinna być kalibrowana.

Magnetometr

Magnetometr mierzy indukcję pola magnetycznego w trzech kierunkach. Ponieważ między biegunami Ziemi przebiegają linie pola magnetycznego, to magnetometr może być użyty do wyznaczenia orientacji obiektu w przestrzeni (jako kompas). Oś magnetometru leżąca w kierunku północ-południe wskazuje wartości maksymalne indukcji, osie prostopadłe wskazują wartości minimalne.

Kluczowym zagadnieniem jest kalibracja magnetometru, który jest obciążony zarówno błędem addytywnym (nazywanym z ang. *hard iron*), jak i multiplikatywnym (nazywanym z ang. *soft iron*). Odczyty magnetometru są wrażliwe na zakłócenia powstałe w wyniku działań innych części układów robotyki. Z tego powodu kalibracja magnetometru jest koniecznością, magnetometr nieskalibrowany jest obciążony błędami uniemożliwiającymi poprawne działanie systemu AHRS.

0.3.2 Kalibracja czujników

W przypadku czujników IMU kalibracja polega na uśrednieniu odczytu n wartości w trakcie spoczynku na równym podłożu. Kolejne odczyty należy korygować o zapisaną wartość średniej.

W przypadku magnetometru w czasie kalibracji należy nim poruszać, tak aby każda oś jego pomiaru obróciła się w przestrzeni o kąt pełny. W przypadku poprawnie skalibrowanego magnetometru pomiary o maksymalnych wartościach powinny mieć na wykresie kształt okręgu ze środkiem w początku układu współrzędnych. W przypadku wystąpienia błędu *hard iron* środek okręgu jest przesunięty, natomiast w wyniku działania błędu *soft iron* następuje zniekształcenie okręgu.

Ważne jest aby kalibracji dokonać w odpowiedniej kolejności, najpierw należy wyznaczyć *hard iron*, a potem *soft iron*. Obliczenie błędu multiplikatywnego wymaga, aby błąd addytywny był już zniwelowany. Błąd *hard iron* jest średnią ze skrajnych wartości n pomiarów, co przedstawiono na wzorze 1, gdzie $\vec{m_n}$ to wektor pomiarów z czujnika.

$$\vec{e_{hard}} = \frac{max\left(\vec{m_n}\right) + min\left(\vec{m_n}\right)}{2} \tag{1}$$

Błąd *soft iron* należy wyznaczyć według wzorów od 3 do 4.

$$\vec{r} = \frac{max\left(\vec{m_n}\right) - min\left(\vec{m_n}\right)}{2} \quad \vec{r} = [r_x, r_y, r_z] \tag{2}$$

$$r_a = \frac{r_x + r_y + r_z}{3} \tag{3}$$

$$\vec{e_{soft}} = \left[\frac{r_a}{r_x}, \frac{r_a}{r_y}, \frac{r_a}{r_z}\right] \tag{4}$$

Ostatecznie odczyty magnetometru należy korygować według wzoru

$$\vec{m_{fixed}} = (\vec{m} - \vec{e_{hard}}) \cdot \vec{e_{soft}} \tag{5}$$

0.3.3 Implementacja obsługi czujników w systemie operacyjnym Linux

Obsługę czujników zdecydowano się oprzeć na systemie Linux, który jest również bazą dla środowiska ROS. Jądro systemu Linux udostępnia dostęp do magistrali komunikacji *I2C* poprzez wywołania systemowe (ang. *system calls*). Urządzenia peryferyjne w systemach pochodnych systemu *UNIX* są reprezentowane przez pliki znakowe. W celu nawiązania połączenia z czujnikiem należy otworzyć plik urządzenia znakowego *I2C* za pomocą wywołania `open()`. Pliki urządzeń peryferyjnych znajdują się w katalogu `/dev/`. Połączenie nawiązuje się wywołaniem `ioctl(device_handle, I2C_SLAVE, i2c_address);`, gdzie `device_handle` to deskryptor otwartego pliku urządzenia znakowego, `I2C_SLAVE` to stała określająca tryb działania urządzania, a `i2c_address` to adres urządzenia podłączonego do magistrali *I2C*. Wymiana z urządzeniami odbywa się poprzez zapis i odczyt danych z pliku urządzenia, za pomocą wywołań `read()` oraz `write`. Istotne jest aby odczytywać i zapisywać odpowiednią liczbę bajtów, oraz poprawnie

skalować odczyty. Informacje na temat formatu przesyłanych danych znajdują się w notach katalogowych czujników. Istnieją dodatkowe biblioteki ułatwiające pracę z urządzeniami *I2C* (np. *SMBUS*), jednak opisany sposób jest najbardziej uniwersalny. Utworzona biblioteka komunikacji z czujnikami *MinIMU-9 v5* będzie działała na każdym urządzeniu z systemem Linux, którego jądro obsługuje komunikację *I2C*.

Program napisano w języku C++ odpowiednio opakowując kod języka C. Błędy zgłaszane przez system *errno* obsłużono przez wyjątki. Adresy rejestrów i urządzeń *I2C* zapisano w silnie typowanych wyrażeniach wyliczeniowych, które parametryzują szablony klas urządzeń magistrali *I2C*. Oprogramowanie obsługi czujników udostępniono w postaci statycznie łączonej biblioteki programistycznej.

0.3.4 Filtr Kalmana

Filtr Kalma jest algorytmem estymacji stanu modelu dynamicznego. Algorytmy tego typu określają wewnętrzny stan obiektu na podstawie pomiarów wejścia i wyjścia tego układu. Filtr wykonuje estymacje rekurencyjnie, na podstawie wcześniejszych obliczeń, a szacunki są optymalne, zakładając że błędy pomiarowe mają rozkład normalny oraz układ jest liniowy. Jeżeli układ dyskretny opisany jest równaniami 6 i 7, gdzie:

$x(t)$ to wektor stanu w funkcji czasu,

$y(t)$ to wektor wyjścia układu (pomiaru) w funkcji czasu,

A to macierz systemowa (przejścia),

B to macierz wejścia,

H to macierz wyjścia (pomiaru),

$v(t), u(t)$ to wektory szumu, kolejno procesu oraz pomiaru.

$$x(t+1) = Ax(t) + Bu(t) + v(t) \tag{6}$$
$$y(t) = Hx(t) + w(t) \tag{7}$$

Działanie filtru Kalmana składa się z dwóch faz: fazy predykcji (*a priori*) oraz aktualizacji (*a posteriori*). Faza predykcji (ekstrapolacji) polega na wyznaczeniu stanu układu w nowej chwili czasu na podstawie modelu układu oraz wcześniejszego stanu. W czasie predykcji wyznaczana jest również nowa macierz niepewności (kowariancji estymacji). Predykcja odbywa się według wzorów 8 i 9, gdzie:

$\hat{x}_a(t)$ to estymacja *a priori* wektora stanu w funkcji czasu,

$\hat{x}_p(t)$ to estymacja *a posteriori* wektora stanu w funkcji czasu,

$P_a(t)$ to macierz kowariancji estymacji stanu *a priori* w funkcji czasu,

Q to macierz kowariancji szumu procesu.

$$\hat{x}_a(t) = A\hat{x}_p(t-1) + Bu(t-1) \tag{8}$$
$$P_a(t) = AP_p(t-1)A^T + Q \tag{9}$$

Kolejnym etapem jest aktualizacja, która odbywa się na podstawie pomiaru wyjścia układu. Ma ona za zadanie poprawić estymację stanu, uwzględniając różnicę między rzeczywistym pomiarem wyjścia, a jego wartością obliczoną *a priori* przy pomocy stanu z fazy predykcji. Kluczowym elementem filtru Kalmana jest *wzmocnienie Kalmana*, które reprezentuje balans ufności w predykcję *a priori* (według modelu) oraz w pomiary wyjścia *a posteriori*. W trakcie fazy aktualizacji, wyznaczane jest także nowe *wzmocnienie Kalmana*. Intuicyjnie można powiedzieć, że wzmocnienie jest wyznaczane tak, by najlepiej balansować pomiędzy przewidywaniem stanu według modelu matematycznego (niedoskonałego, a także nie obejmującego zakłóceń procesu) oraz przewidywaniem według pomiaru (obarczonego szumem pomiarowym). Faza aktualizacji odbywa się zgodnie ze wzorami od 10 do 12, gdzie:

$K(t)$ to *wzmocnienie Kalmana* w funkcji czasu,

$\hat{P}_p(t)$ to macierz kowariancji estymacji stanu *a posteriori* w funkcji czasu,

R to macierz kowariancji szumu pomiarowego.

$$K(t) = P(t) H^T \cdot \left(H P_a(t) H^T + R\right)^{-1} \tag{10}$$

$$\hat{x}_p(t) = \hat{x}_a(t) + K \cdot (y(t) - H\hat{x}_a(t)) \tag{11}$$

$$P_p(t) = (I - K(t) H) \cdot P_a(t) \tag{12}$$

0.3.5 Zastosowanie filtru Kalmana w fuzji danych pochodzących z czujnika IMU

Filtr Kalmana można zastosować, aby oszacować orientację obiektu w przestrzeni na podstawie odczytów z czujnika IMU. Fuzja danych przy pomocy filtru pozwala zniwelować niedoskonałości żyroskopu (dryft) oraz akcelerometru (szumy pomiarowe). Zazwyczaj dane z żyroskopu są używane w roli wejścia systemu, na etapie predykcji, a pomiary z akcelerometru są wykorzystywane podczas fazy aktualizacji. Czujniki IMU pozwalają jedynie na określenie kątów *roll* oraz *pitch*, w dalszej części raportu zostanie opisany sposób wyznaczenia kąta *yaw* na podstawie odczytów magnetometru.

Dane wejściowe filtru Kalmana

Na podstawie danych z akcelerometru można obliczyć dwa z kątów opisujących orientację obiektu w przestrzeni (wykorzystując fakt, stałego przyspieszenia ziemskiego w pionowej osi inercyjnego układu odniesienia). Kąty *roll* i *pitch* są wyznaczane według wzorów 13 oraz 14, gdzie gdzie: ϕ i θ to kolejno kąty *roll oraz pitch*, a a to odczyty akcelerometru.

$$\tilde{\phi}_a = arctan\left(\frac{a_y}{\sqrt{a_x^2 + a_z^2}}\right) \cdot \frac{180}{\pi} \tag{13}$$

$$\tilde{\theta}_a = arctan\left(\frac{a_x}{\sqrt{a_y^2 + a_z^2}}\right) \cdot \frac{180}{\pi} \tag{14}$$

Prędkości kątowe odczytane z żyroskopu nie mogą być użyte wprost, ze względu na powiązanie osi odczytu w układzie odniesienia. Obrót układu wokół jednej z jego osi może powodować

zmianę orientacji obiektu w wielu osiach zewnętrznego układu odniesienia. Transformacji prędkości kątowych z układu żyroskopu do układu inercyjnego należy dokonać według wzoru 15, gdzie: ϕ, θ, ψ to kolejno kąty *roll, pitch oraz yaw*, a g to odczyty żyroskopu.

$$\begin{bmatrix} \dot{\phi_g} \\ \dot{\theta_g} \\ \dot{\psi_g} \end{bmatrix} = \begin{bmatrix} 1 & sin\phi tan\theta & cos\phi tan\theta \\ 0 & cos\phi & -sin\phi \\ 0 & sin\phi sec\theta & cos\phi sec\theta \end{bmatrix} \cdot \begin{bmatrix} g_x \\ g_y \\ g_z \end{bmatrix} \tag{15}$$

Równania filtru Kalmana dla fuzji danych

Na podstawie opisu teoretycznego filtru Kalmana oraz przygotowanych danych z czujników można przygotować ostateczną wersję systemu podstawiając konkretne macierze systemowe i wektory z równań: 16 i 17 do równań od 6 do 12. Symbol Δt oznacza okres próbkowania czujników.

$$A = \begin{bmatrix} 1 & -\Delta t & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & -\Delta t \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad B = \begin{bmatrix} \Delta t & 0 \\ 0 & 0 \\ 0 & \Delta t \\ 0 & 0 \end{bmatrix} \quad H = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \tag{16}$$

$$\vec{x}(t) = \begin{bmatrix} \hat{\phi}(t) \\ b_{\hat{\phi}(t)} \\ \hat{\theta}(t) \\ b_{\hat{\theta}(t)} \end{bmatrix} \quad \vec{u}(t) = \begin{bmatrix} \dot{\phi_g}(t) \\ \dot{\theta_g}(t) \end{bmatrix} \quad \vec{y_t} \begin{bmatrix} \tilde{\phi_a}(t) \\ \tilde{\theta_a}(t) \end{bmatrix} \tag{17}$$

Macierze Q (kowariancja szumu procesu) oraz R (kowariancja szumu pomiaru) są dobierane przez użytkownika. Są to zazwyczaj macierze diagonalne, im większa wartość na ich przekątnej, tym większe zakłócenia w danej części układu. Początkowa wartość na przekątnej P reprezentuje wstępną ufność w dokładność modelu.

0.3.6 Algorytm kompasu oraz kompensacja przechylenia

Zgodnie z wcześniejszym opisem magnetometru można użyć jako kompasu w celu wyznaczenia kąta *yaw*. Jednak w przypadku przechylenia układu czujników w pozostałych kątach płaszczyzna kompasu (magnetometru) zostanie odchylona od płaszczyzny podłoża, co spowoduje zakłamanie odczytów. Aby zniwelować ten problem można użyć wyliczonych uprzednio kątów *roll* oraz *pitch*. Służy do tego formuła *kompensacji przechylenia* (ang. *tilt compensation*, którą przedstawia wzory od 18 do 20.

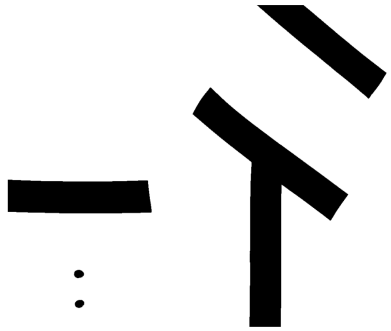
$$h_x = m_x cos\hat{\theta} + m_y sin\hat{\theta} sin\hat{\phi} + m_z sin\hat{\theta} cos\hat{\phi} \tag{18}$$

$$h_y = m_y cos\phi - m_z sin\phi \tag{19}$$

$$\psi = atan2(-h_y, h_x) \tag{20}$$

0.3.7 Implementacja programistyczna systemu AHRS

System AHRS zaimplementowano w języku C++ w postaci biblioteki programistycznej. Aby umożliwić jak najszersze zastosowanie opracowanego kodu zadbano o jego wysoką przenaszalność. Aby, umożliwić działanie programu na urządzaniach wbudowanych *bare metal* w programie unikano polegania na wyjątkach oraz dynamicznej alokacji pamięci. Większość klas i funkcji jest parametryzowana przez szablony w czasie kompilacji, co pozwala na generowanie szybkiego kodu (kosztem hipotetycznego zwiększenia rozmiaru plików wykonywalnych).



Rysunek 1: Przykładowy labirynt poddany binaryzacji

0.4 Odometria robota

0.5 System wizyjny

0.5.1 Sprzęt wizyjny

Jako bazę do wyznaczania ścieżki robota w terenie zdecydowano się użyć kamery *Raspberry Pi Camera V2*, która oferuje obraz wideo w rozdzielczości *full HD* oraz zdjęcia w rozdzielczości do *4K*. Kamera jest wyposażona we wbudowany obiektyw szerokokątny, co jest korzystne w zastosowaniach robotyki.

0.5.2 Wyznaczanie przestrzeni ruchów robota

Kamerę przygotowano do wykorzystania w celu wykonywania zdjęcia labiryntu, w którym poruszać może się robot. Obraz z kamery należy przetworzyć aby uzyskać mapę przestrzeni, po której może poruszać się robot. Biała przestrzeń na mapie powinna oznaczać miejsca, w których może znaleźć się środek robota, czarne pola to miejsca zabronione (przeszkody oraz ich margines). Budowę mapy wykonano w następujących krokach:

- binaryzacja zdjęcia z kamery,
- wykrycie położenia robota oraz jego promienia na podstawie markerów,
- usunięcie obrysu robota z przestrzeni zabronionej,
- naniesienie marginesu przeszkód.

Ze względu na zdalny charakter pracy w semestrze letnim 2020 przygotowano prototypowy model labiryntu. Podstawą przetwarzania obrazu labiryntu jest binaryzacja metodą *isodata*. Oparcie wizji na algorytmie binaryzacji wymaga odpowiedniego kontrastu zdjęcia oraz dobrego oświetlenia labiryntu. *Isodata* jest adaptacyjną metodą binaryzacji zdjęć, która gwarantuje, że wartość progowania jest średnią ze średnich wartości ekspozycji podzielonych fragmentów zdjęcia. Taki algorytm binaryzacji odpowiada algorytmowi klasteryzacji *k-średnich*, gdzie k wynosi dwa. Przykładowy labirynt po binaryzacji przedstawiono na rysunku 1.

(a) Wykryte markery i ich centroidy (b) Labirynt po usunięciu markerów

Rysunek 2: Wykrywanie oraz usuwanie markerów z mapy labiryntu

W następnym kroku algorytm wizyjny przeprowadza wykrycie pozycji i rozmiaru robota na podstawie dwóch czerwonych markerów umieszczonych na jego obwodzie. Punkty o charakterystycznym kolorze najlepiej wykrywać w przestrzeni barw HSV (barwa, nasycenie, jasność ang. *hue, saturation, value*). W omawianej przestrzeni kolory są reprezentowane przez wartość kąta; od zera do kąta pełnego, przy czym wartości reprezentuje często się jako ułamek dziesiętny z maksymalnej wartości. Kolor czerwony jest w takim systemie opisywany przez kąty zbliżone do zera. Markery zdecydowano się kwalifikować w przedziale odcieni czerwonego od 0,9 do 1 oraz 0 do 0,1. Wartości nasycenia oraz jasności markerów są akceptowane dla wartości większych od 0,3. Po progowaniu pozycji markerów dokończono proces ich segmentacji przez etykietowanie oraz obliczenie parametrów obszaru zajmowanego przez markery. Wyznaczono ich centroidy; punkt między centroidami markerów to środek robota. Po wyznaczeniu pozycji robota można usunąć jego kształt z mapy przez odjęcie maski jego obrysu z obrazu. Wyniki omówionych operacji przedstawiono na rysunkach 2a oraz 2b.

Ostatni etap przygotowania mapy polega na dodaniu do ścian labiryntu marginesu, który zapewni, że krawędź robota nie zetknie się z przeszkodami. W tym celu dokonano procesu erozji binarnej przestrzeni dozwolonej ruchu robota. Margines ścian wyznaczono na podstawie średnicy robota wyznaczonej przy segmentacji markerów. Erozji dokonano używając jądra w kształcie okręgu o promieniu równym ponad połowie promienia robota. Ostatecznie powstaje mapa binarna, na której kolor biały oznacza przestrzeń w której może znaleźć się środek robota, przedstawiono ją na rysunku 3

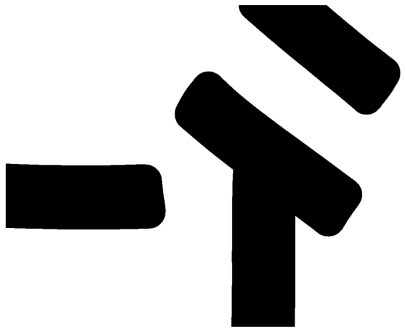
Do wykonania wymienionych operacji użyto funkcji z biblioteki *scikit-image* dla języka Python.

0.6 Analiza biznesowa

W ramach projektu dokonano analizy możliwości biznesowych związanych z robotyką mobilną w regionie Górnego Śląska. Pełną treść analizy zamieszczono w osobnym dokumencie.

Przegląd dostępnych źródeł finansowania

Przeanalizowano ofertę dofinansowania projektów oraz przedsięwzięć biznesowych: zarówno krajowych, jak i zagranicznych. W ramach przeglądu wzięto pod uwagę oferty aktualne oraz



Rysunek 3: Przykładowy labirynt z naniesionym marginesem ścian

niedawno zakończone. Dofinansowania w ramach programów rozwoju oraz inkubatorów charakteryzowały się atrakcyjnymi ofertami, niektóre z nich nie wymagały wkładu własnego, inne uzależniały go od wysokości kwoty pomocy. Wkład własny większości rozpatrywanych programów nie przekraczał parunastu procent. Oferty proponowały wsparcie w wysokości od parudziesięciu do paruset tysięcy złotych. Częstym wymogiem uczestnictwa w programie była gotowości prototypu produktu. Wsparcie w projektach jest ograniczone do małych firm, przy czym pożądana jest współpraca z uczelnią. W ramach przeglądu analizowano programy takie jak: *POWER*, *Program Akceleracyjny KPT* oraz *Program Inteligentny Rozwój*.

Przykłady rozwijających się firm na rynku robotyki

Spośród polskich działalności przeanalizowano rozwój firm takich jak *KP Labs* oraz *Future Processing*. Zwrócono uwagę na źródła finansowania projektów. Produkty o wysokiej innowacyjności i aspekcie naukowym (takie jak w przypadku firmy *KP Labs*) uzyskały bardzo wysokie dofinansowania unijne, przekraczające połowę wartości poszczególnych przedsięwzięć. Produkty związane na przykład z kosmonautyką mogły liczyć na dotacje wielkości milionów złotych.

Zdecydowano się także na analizę zagranicznych firm, które w ostatnich latach odniosły sukces na arenie międzynarodowej. Zwrócono uwagę na działalność skandynawskiej firmy *Antmicro*, która posiada polskie filie. Wiele z wykorzystywanych przez firmę technologii należy do grupy *open source* oraz *open source hardware*. Ciekawym przypadkiem okazała się inicjatywa *CommaAI*, której twórcą jest znany programista Georg Hotz. Jego firma skupia się na wyposażaniu istniejących samochodów, w systemy jazdy autonomicznej. *CommaAI* oferuje oprogramowanie oraz moduły współpracujące z czujnikami i kamerami wbudowanymi w samochody oraz smartfony. Jest to ciekawe podejście wykorzystujące wykosi poziom cyfryzacji otaczających technologii. Takie podejście ułatwia tworzenie produktów oraz pozwala na oferowanie ich w konkurencyjnych cenach.

Analiza SWOT potencjalnego przedsięwzięcia

W ramach analizy przeprowadzono analizę *SWOT* hipotetycznego przedsięwzięcia w dziedzinie robotyki mobilnej w warunkach biznesowych Górnego Śląska. Analiza skupia się na możliwościach założenia działalności biznesowej przez absolwentów Politechniki Śląskiej.

Mocne strony

- potencjalni członkowie zespołu, tzn. absolwenci kierunków technicznych są zapoznani z bieżącymi, nowoczesnymi technologiami,
- potencjalni członkowie zespołu rozumieją potrzeby nowoczesnego przemysłu.
- dobry wewnętrzny kontakt absolwentów z uczelnią.

Słabe strony

- potencjalni członkowie zespołu (absolwenci, doktoranci) mają małe doświadczenie biznesowe i menadżerskie.

Szanse

- zainteresowanie medialne i społeczne nowymi technologiami,
- liczne projekty dofinansowania; unijne oraz krajowe,
- możliwość współpracy z jednostkami naukowymi i uczelniami badawczymi.

Zagrożenia

- nieprzygotowanie polskiego rynku na adaptowanie nowych technologii,
- prawdopodobny kryzys,
- ryzyko przejęcia firmy przez zagraniczny koncern, ze względu na brak bezpośredniego odbiorcy zaawansowanych produktów na rodzimym rynku.

0.7 Podsumowanie

Spis rysunków

Spis tablic

Spis listingów

Bibliografia