Politechnika Warszawska

WYDZIAŁ MECHANICZNY
ENERGETYKI I LOTNICTWA

Instytut Techniki Lotniczej i Mechaniki Stosowanej

# System do planowania trajektorii obiektu na podstawie dwuwymiarowej mapy otoczenia

Michał Modzelewski

Numer albumu: 292768

Warszawa, 2022

# Spis treści

# 1 Wstęp

Celem projektu jest stworzenie systemu do bezkolizyjnego planowania trajektorii robota poruszającego się w zamkniętym pomieszczeniu. W celach optymalizacyjnych system został rozproszony i podzielony na podsystem do symulacji egzekucji trajektorii, znajdujący się lokalnej maszynie oraz podsystem do wyznaczania trajektorii, który stanowi wirtualna maszyna wraz ze stosownym oprogramowaniem.

System został zrealizowany przy wykorzystaniu funkcjonalności Microsoft Azure Portal:

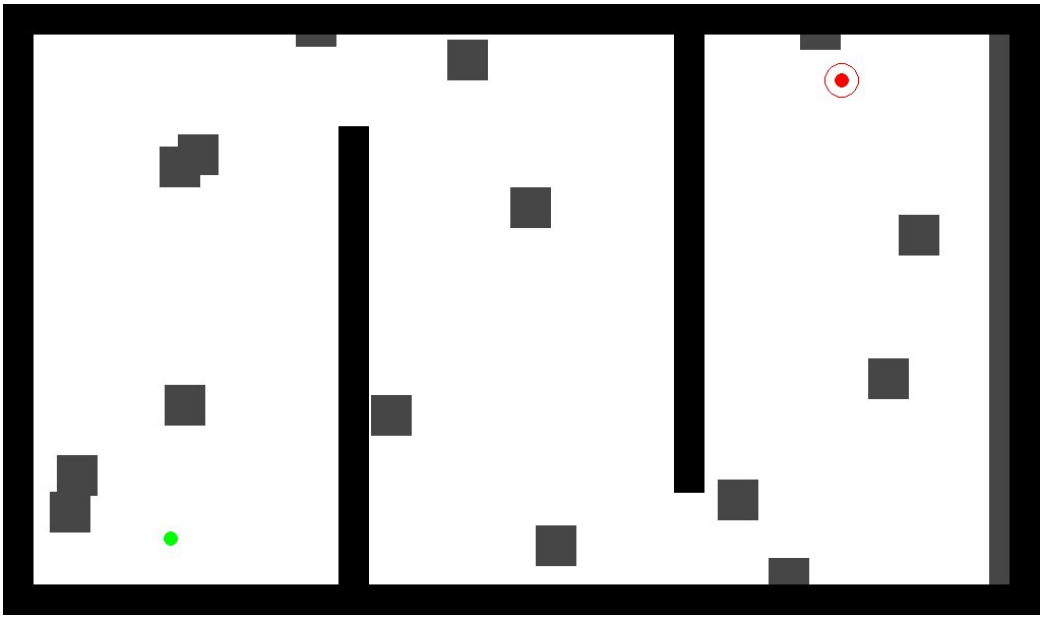
- Wirtualna maszyna Linux (Ubuntu 20.04) - obliczenia
- Konta magazynu - wymiana danych pomiędzy lokalną a wirtualną maszyną

Całe oprogramowanie zostało napisane w języku Python 3.6 z wykorzystaniem modułów: azure, pygame, pickle, yaml, os, pathlib, time, random, math, collections.

# 2 Wyznaczanie trajektorii

## 2.1 Model obiektu i otoczenia

W projekcie robot dla którego wyznaczana jest trajektoria został za modelowany jako punkt materialny. Znajduje się on w zamkniętym pomieszczeniu z przeszkodami - wymiary pomieszczenia 20x34 m. Ze względu na fakt, że robot porusza się po powierzchni, model otoczenia uproszczono do dwuwymiarowej mapy widocznej na Rysunku 1.



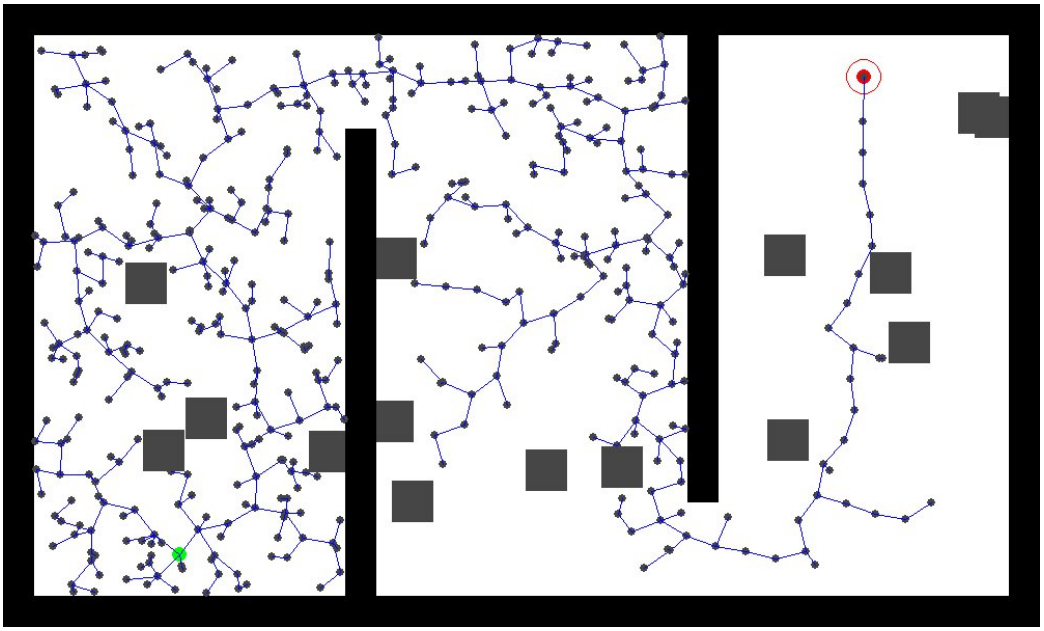
Rysunek 1: Dwuwymiarowy model otoczenia.

Kontury zaznaczone czarnym kolorem oznaczają ściany pomieszczenia, w którym znajduje się robot, w związku z czym ten rodzaj przeszkody nie zmienia swojego położenia. Na rysunku widoczne są również szare kwadraty, które symbolizują przeszkody zmieniające swoje położenie np. ludzie znajdujący się w pomieszczeniu lub inne roboty. Dobór położenia tych przeszkód jest losowy. Przyjęto założenie, że podczas wyznaczania trajektorii system posiada informację na temat położenia przeszkód oraz robota. Punkt startowy oznaczono kolorem zielonym, a punkt docelowy czerwonym.

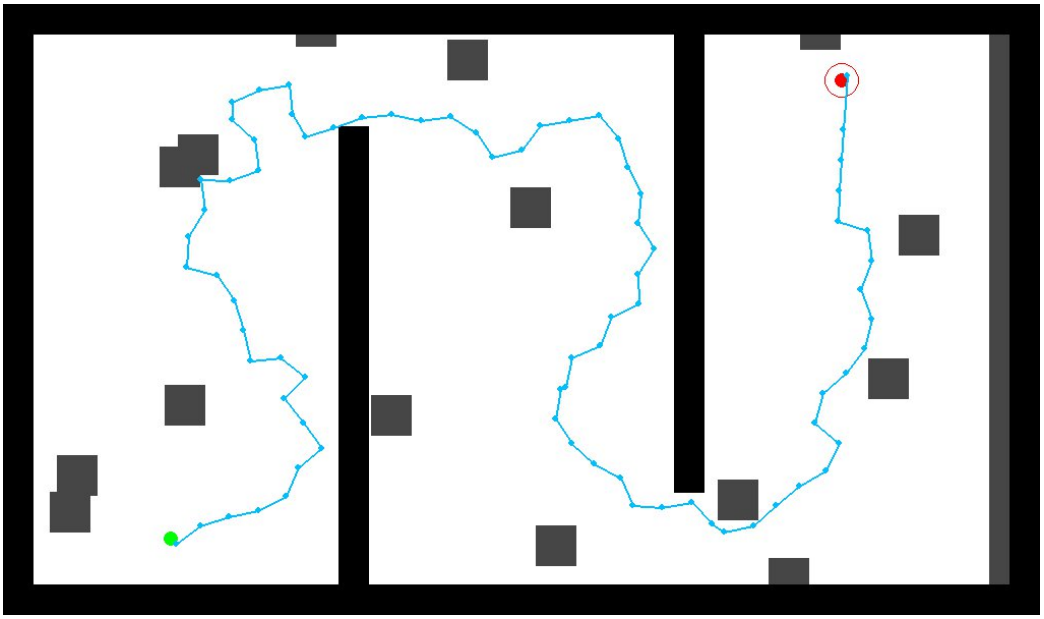
## 2.2 Algorytm

Podsystem do wyznaczania trajektorii, korzysta z algorytmu RRTOD (ang. Rapidly-exploring Random Tree - Optimized Dijkstra), którego szczegółowe wyjaśnienie zasady działa oraz uzasadnienie jego wyboru zostało przedstawione w pracy "Dobór algorytmu do systemu planowania trajektorii obiektu na podstawie dwuwymiarowej mapy otoczenia"[3].

Algorytm RRTOD to połączenie dwóch algorytmów: RRT (ang. Rapidly-exploring Random Tree) oraz Dijkstra. Pierwszy z nich służy do efektywnego znajdywania trajektorii poprzez losowe próbkowanie otoczenia. Wynikiem działania tego algorytmu jest graf, w którym występuje połączenie pomiędzy punktem startowy i docelowy.



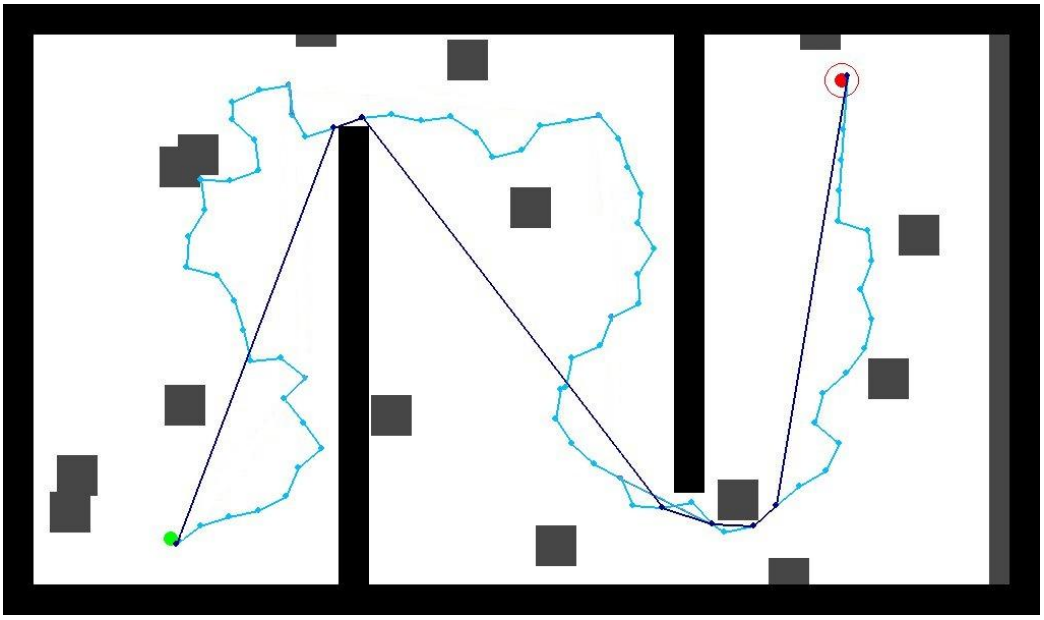
Rysunek 2: Całkowity graf algorytmu RRT



Rysunek 3: Trajektoria uzyskana z algorytmu RRT.

Kolejnym etapem działania algorytmu RRTOD jest użycie algorytmu Dijkstra w celu optymalizacji uzyskanej trajektorii pod względem długości. Nowo

powstała trajektoria składa się z połączeń między węzłami trajektorii RRT, które zapewniają minimalną długość drogi z punktu startowego do docelowego.



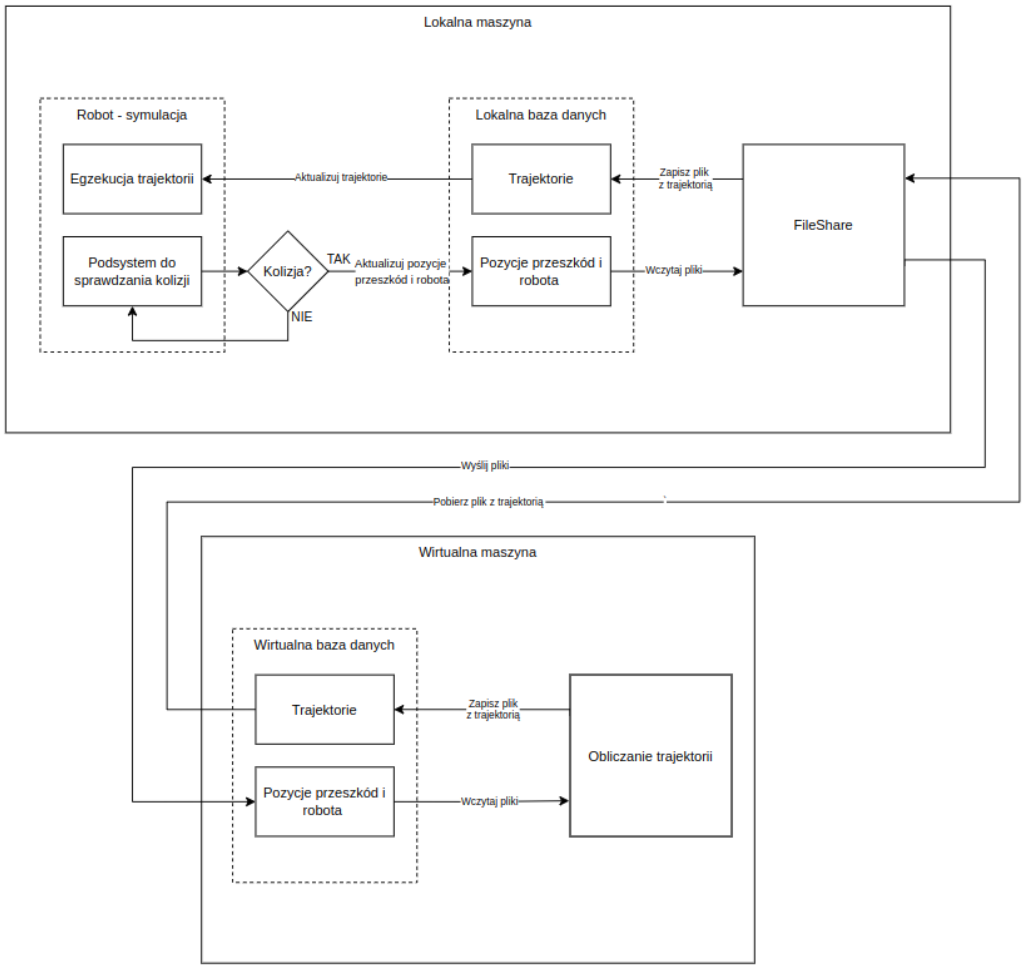
Rysunek 4: Trajektorie uzyskane przez algorytmy: RRT (jasnoniebieski kolor), RRTOD (ciemnoniebieski kolor).

Tak uzyskana trajektoria jest efektem działania podsystemu do wyznaczania trajektorii.

# 3 Opis systemu

Projektowany system jest rozproszony i składa się się z podsystemu do symulacji egzekucji trajektorii robota, znajdującym się na lokalnej maszynie i podsystemu do wyznaczania trajektorii, znajdujący się na wirtualnej maszynie. Na lokalnej maszynie sprawdzane jest dodatkowo czy aktualnie wykonywana trajektoria nie jest kolizyjna. Jeżeli tak, to w lokalnej bazie danych zapisywane są położenia przeszkód oraz aktualna pozycja robota. Następnie dane są wysyłane do wirtualnej maszyny za pomocą obiektu FileShare, który pozwala na komunikacje pomiędzy maszynami. Gdy wirtualna maszyna wykryje nowe pliki z położeniami przeszkód i robota, uruchamiany jest program do wyznaczania nowej trajektorii na podstawie otrzymanych danych. W kolejnym kroku nowa trajektoria jest zapisywana do wirtualnej

bazy danych, skąd za pomocą obiektu FileShare jest pobierana do lokalnej bazy danych. Następnie trajektoria robota jest aktualizowana. Proces się powtarza do momentu dotarcia robota do miejsca docelowego.



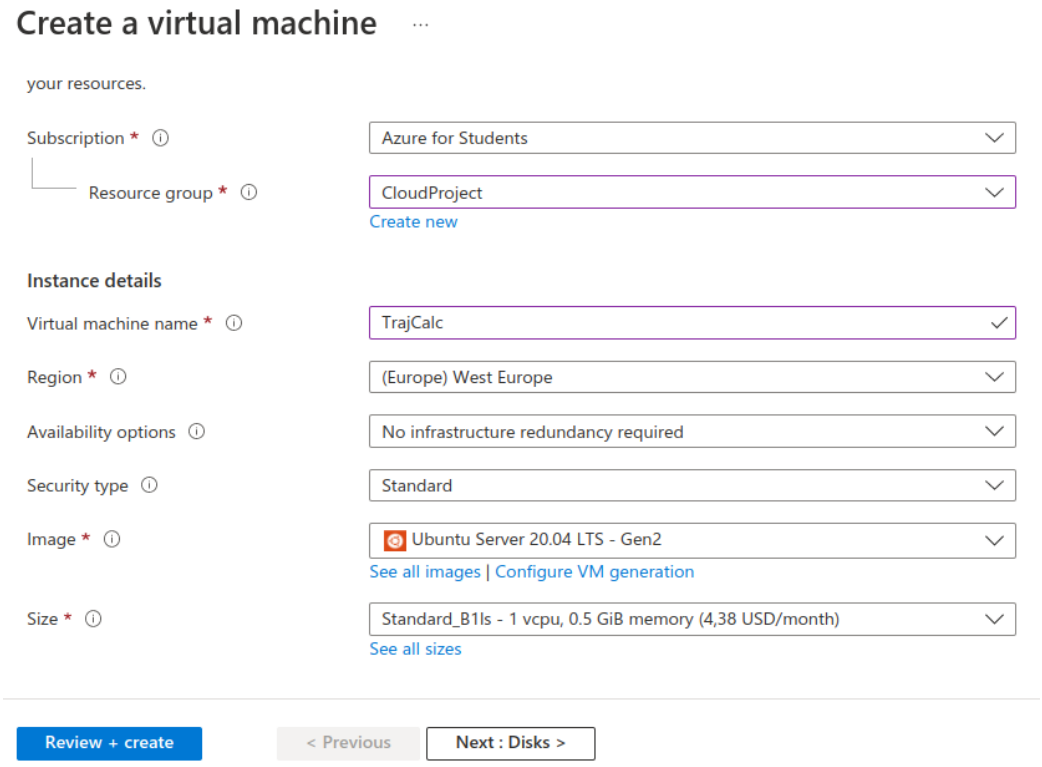

Rysunek 5: Schemat systemu

# 4 Uruchomienie systemu

W celu uruchomienie systemu konieczne jest założenie konta na Microsoft Azure Portal [1]. Następnie konieczne jest stworzenie wirtualnej maszyny i konta magazynu (ang. Storage account), a następnie połączenie ich ze sobą. Ostatnim krokiem jest wgranie oprogramowania na wirtualną maszynę oraz uruchomienie programu.

## 4.1 Tworzenie wirtualnej maszyny

Aby stworzyć nową instancję wirtualnej maszyny należy wyszukać w Microsoft Azure "Maszyny wirtualne". Po kliknięciu ikony ukaże się lista dostępnych maszyn wirtualnych. Następnie należy kliknąć "Dodaj". W kreatorze nowej maszyny należy wybrać opcje tak jak na poniższym rysunku.




Rysunek 6: Kreator wirtualnej maszyny

W celu przyspieszenia obliczeń zaleca się wybranie maszyny o większej mocy obliczeniowej. Reszta ustawień może pozostać domyślnie. Aby utworzyć wirtualną maszynę należy w ostatniej zakładce kliknąć "Utwórz".

Podczas tworzenia wirtualnej maszyny powinien pobrać się klucz prywatny, który należy zapisać w katalogu w którym przechowywane są pliki związane z projektem.

Aby połączyć się z wirtualną maszyną, w terminalu (Ubuntu) należy otworzyć katalog, w którym zapisany został klucz prywatny i wykonać dwie komendy:

$chmod400azureuser.pem$

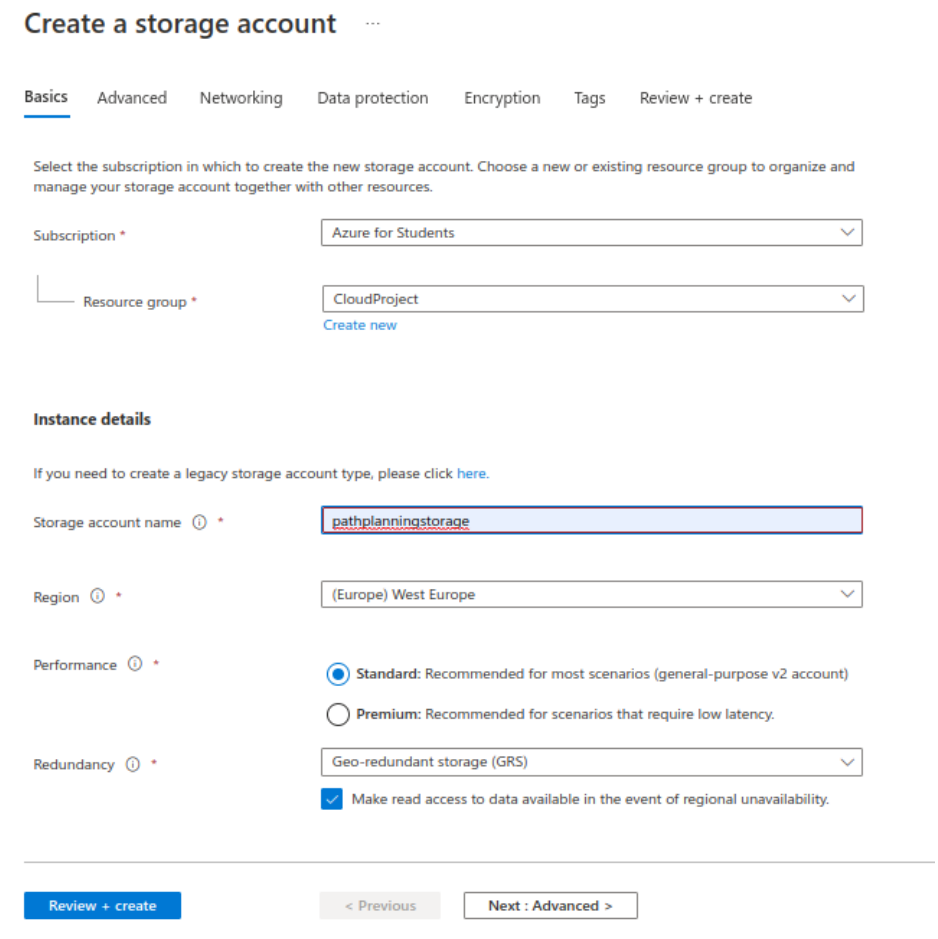
$ssh-i < nazwaprywatnegoklucza > azureuser@ < ipwirtualnejmaszyny >$

Rezultatem powyższych działań powinno być uzyskanie połączenia z wirtualną maszyną.

Kolejnym krokiem jest instalacja (na wirtualnej maszynie) interpretera języka Python oraz modułów: azure, pygame, pickle, yaml, os, pathlib, time, random, math, collections.

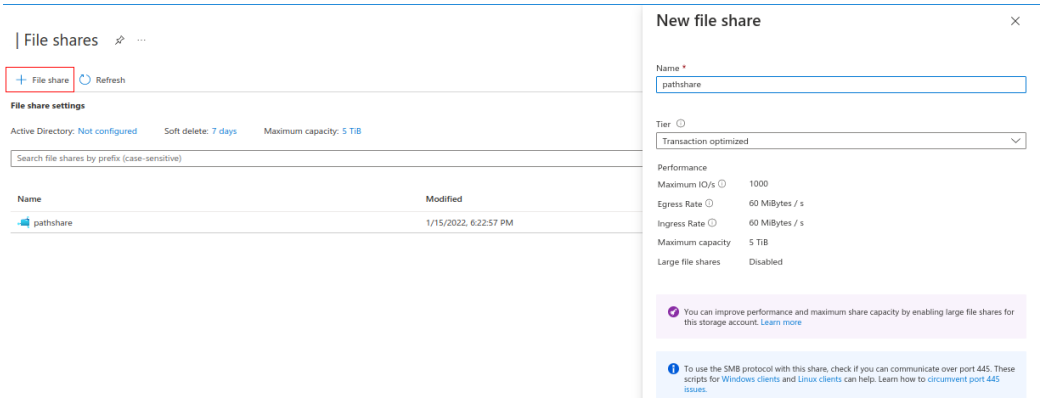
## 4.2 Tworzenie konta magazynu

W celu stworzenia nowego konta magazynu należy wyszukać w Microsoft Azure "Konta magazynu"(ang. Storage accounts). Następnie należy kliknąć przycisk "Utwórz", po czym ukaże się kreator, w którym należy wprowadzić ustawienia, jak na Rysunku 7.



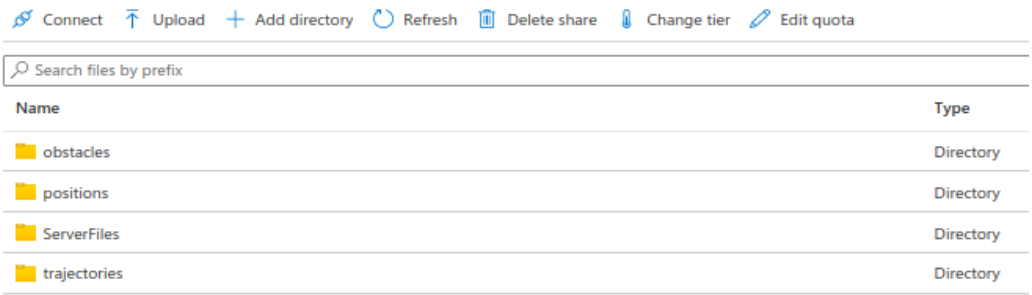

Rysunek 7: Kreator konta magazynu

Po stworzeniu konta magazynu, należy wybrać "Udziały plików"i dodać "Udział plików"tak jak na Rysunku 8




Rysunek 8: Kreator udziałów plików

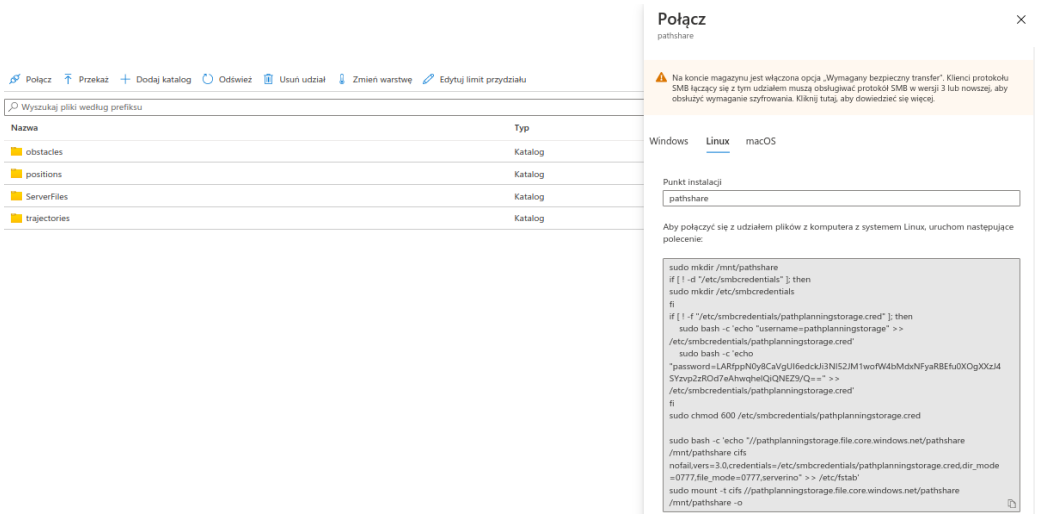
Po wykonaniu powyższych instrukcji należy w stworzonym udziale plików utworzyć katalogi tak jak na Rysunku 9.




Rysunek 9: Katalogi udziałów plików

Do folderu ”ServerFiles” należy wgrać pliki z katalogu o tej samej nazwie z repozytorium [2].

Aby połączyć pliki z wirtualną maszyną, należy kliknąć ”Połącz” oraz wybrać metodę ”SSH”. Następnie skopiować polecenie i uruchomić je na wirtualnej maszynie - Rysunek 10.




Rysunek 10: Łączenie plików z wirtualną maszyną

## 4.3 Uruchamianie programu

W celu uruchomienia programu na lokalnej maszynie należy pobrać folder "LocalFiles" z repozytorium [2] oraz z katalogu $LocalFiles/scripts$ uruchomić w terminalu plik $TrajExeSim.py$ za pomocą polecenia:

$python3\ TrajExeSim.py$

Na wirtualnej maszynie należy uruchomić plik $PathCalcSever.py$ za pomocą polecenia:

$python3\ /mnt/pathshare/TrajExeSim.py$

Rezultatem powyższych operacji powinna być animacja ilustrująca zmianę pozycji robota w celu dostania się do pozycji docelowej.

# 5 Podsumowanie

Zrealizowany projekt pozwolił na zaznajomienie się działaniem i implementacją systemów rozproszonych. Wykorzystanie wirtualnej maszyny do złożonych obliczeń może pozwolić na odciążenie komputera pokładowego robota, a co za tym idzie przyspieszyć działanie systemu. Należy jednak pamiętać, że w przypadku obliczeń rozproszonych trzeba się liczyć z opóźnieniem wynikającym z przesyłu danych. Niemniej jednak przy zastosowaniu wirtualnej maszyny o dużej mocy obliczeniowej, czas potrzebny na wyznaczenie nowej trajektorii powinien być o wiele krótszy niż przy wykonywaniu obliczeń tylko na lokalnej maszynie.

# 6 Literatura

[1] *Microsoft Azure Portal.* https://azure.microsoft.com/en-us/features/azure-portal/.

[2] Michał Modzelewski. *https://github.com/MicMod/PathPlanning.*

[3] Michał Modzelewski. *Dobór algorytmu do systemu planowania trajektorii obiektu na podstawie dwuwymiarowej mapy otoczenia.* 2022.