Inteligentny System Zarządzania Temperaturą

Spis treści

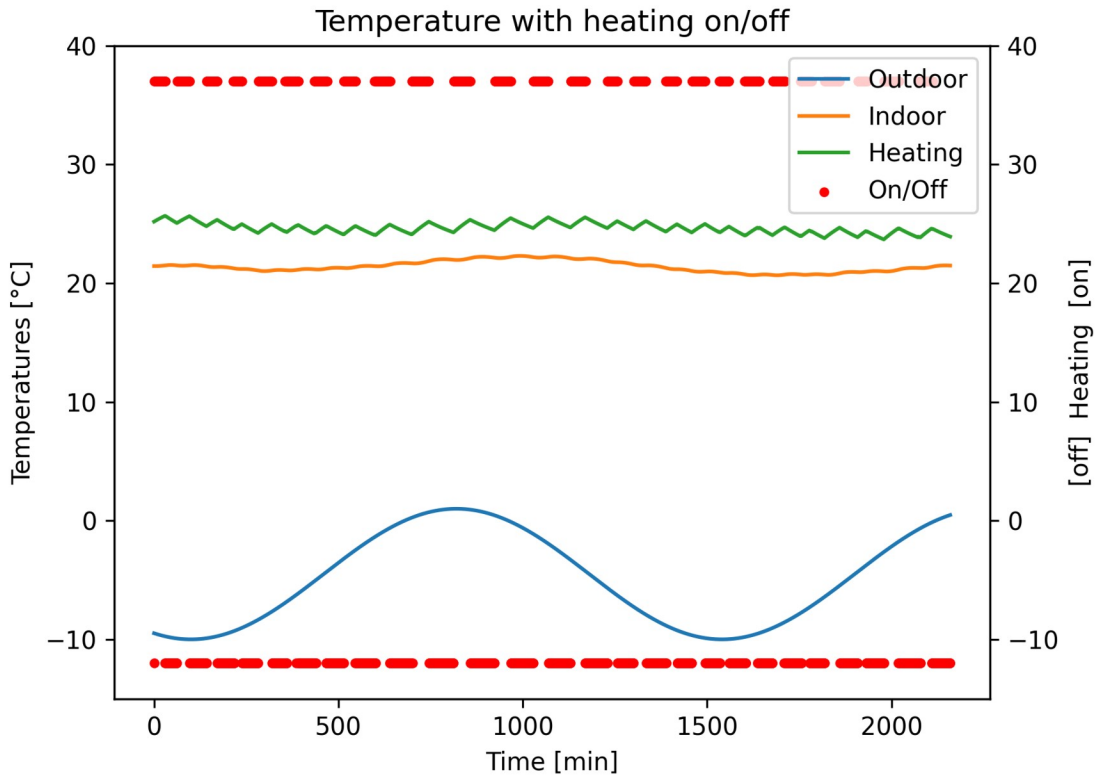
UWAGA:

Jest to pierwsza wersja projektu, która wymaga jeszcze pracy w zakresie symulacji środowiska jak i modelu A3C.

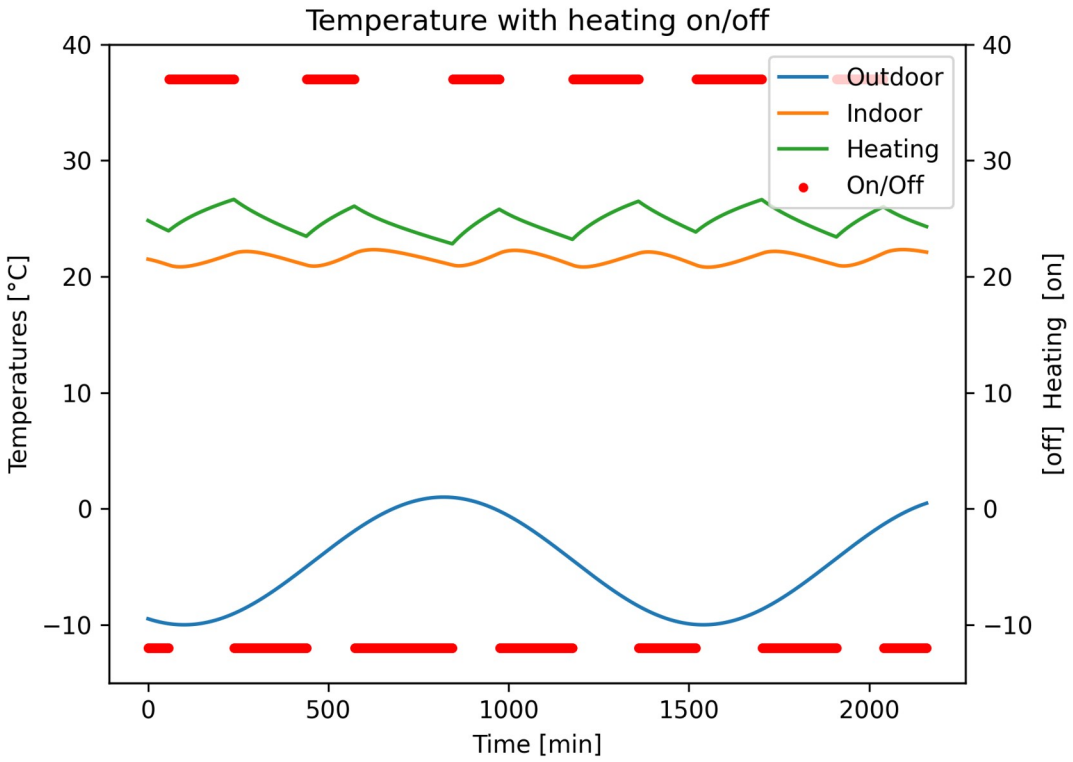
Opis projektu

Projekt[+] ma na celu stworzenie modelu AI do systemu zarządzania temperaturą w budynku z wykorzystaniem technik uczenia maszynowego (Deep Reinforcemend Learning), a konkretnie modelu A3C (Asynchronous Advantage Actor-Critic). System ma za zadanie automatycznie regulować ogrzewanie, aby utrzymać optymalną temperaturę określoną przez użytkownika, jednocześnie minimalizując częstość przełączania ogrzewania i odchylenie od zadanej temperatury. Projekt umożliwia symulowanie wymiany ciepła między budynkiem a otoczeniem, ale wierne odzwierciedlenie fizyki tego procesu nie było (na razie) głównym przedmiotem tego projektu.

Dla celów porównawczych powstał prosty model dwustanowy, który włącza ogrzewanie gdy temperatura spada poniżej ustalonego poziomu (histereza 1°C). Porównanie tych modeli w przygotowanym środowisku możliwe jest dzięki wygenerowanym danym:



Powyższy wykres przedstawia sterowanie temperaturą (otwarcie/zamknięcie zaworu) przez model A3C. Został on wygenerowany z pomocą biblioteki Matplotlib (Python). Zaprezentowano temperaturę na zewnątrz (outdoor), wewnątrz budynku (indoor) oraz temperaturę podłogi (heating). Czerwone kropki oznaczają ogrzewanie włączone, kiedy znajdują się u góry wykresu oraz wyłączone, kiedy znajdują się na dole wykresu.
Wykres temperatur dla prostego sterowania dwustanowego zaprezentowano poniżej:



Wykres przedstawia działanie systemu w tej samej sytuacji, ale sterowaniem temperaturą zarządza prosty mechanizm dwustanowy z histerezą 1°C.

Wyliczone na podstawie danych tabelarycznych* odchylenie standardowe dla zadanej 21,5°C kształtuje się podobnie dla obu model, to znaczy na poziomie ~0,47°C. Porównując wykresy można jednak zauważyć, że model A3C reguluje temperaturę wewnątrz budynku zapewniając rzadsze i mniej odczuwalne odchylenia.

+ Projekt dostępny jest na platformie GitHub pod następującym linkiem:
https://github.com/daniel-w-p/a3c_heating_controller

* Wspomniane dane tabelaryczne dostępne są pod następującym linkiem:
https://github.com/daniel-w-p/a3c_heating_controller/tree/main/data/table

Co udało się osiągnąć:

Wyszkolony model przełącza ogrzewanie pomiędzy stanem włączonym lub wyłączonym (np. zawór dwu-stanowy). Model umożliwia analizę sytuacji (stanu) dla dowolnej liczby pomieszczeń o zbliżonej charakterystyce i określeniu najlepszej akcji (włącz / wyłącz). Model operuje na danych zebranych z 7 godzin w 10-minutowych odstępach (42 wektory). Dane te są traktowane jako aktualny stan zgodnie z procesem decyzyjnym Markova.
(To podejście ma niewielki skutek uboczny: przez pierwsze 7 godzin model operuje na niepełnych danych, jednak nie powoduje to znaczącego obniżenia jakości predykcji)

W ramach projektu powstały cztery główne elementy:

- Model środowiska wykorzystywany do uczenia modelu A3C oraz do symulacji efektów działania modelu w środowisku (wymiana ciepła między budynkiem a otoczeniem). Wartości temperatur są wyznaczane na podstawie równań różniczkowych przybliżanych numerycznie.
- Model Deep Reinforcemend Learning – A3C – model wykorzystywany do sterowania temperaturą. Jest uczony na modelu środowiska.
- Symulator – graficznie prezentuje informacje o temperaturach dla środowiska oraz budynku składającego się z 4 pomieszczeń.
- ‘Tryb cichy’ – skrypt napisany w Pythonie, który tworzy dla zadanej temperatury dane tabelaryczne, wykresy oraz wyznacza odchylenie standardowe.

Wszystkie wymienione komponenty zostały opisane w dalszej części dokumentu.

Kolejne planowane etapy

- Dopracować symulator, żeby lepiej odzwierciedlał rzeczywistość (wymiana ciepła, otwarte okna, nagłe zmiany pogody).
- Spróbować zmniejszyć model i zachować dokładność.
- Zwiększyć model (np. liczbę warstw) by uzyskać większą dokładność.
- Przeszkolić model, który będzie regulował ogrzewanie w sposób płynny (procent otwarcia: 0-100%).

Komponenty

Projekt składa się z czterech kluczowych komponentów:

Moduł Symulacji Środowiska

Moduł ten, oparty na równaniach różniczkowych symuluje dynamikę temperatury w budynku. Dane o temperaturach i stanie ogrzewania są przekazywane do środowiska pracy modelu, symulatora lub cichego trybu. Ten ostatni zapisuje dane do pliku oraz generuje wykres.

Równania różniczkowe modelu

Równanie opisujące zmianę temperatury w pomieszczeniu w czasie jest zdefiniowane jako:

$$\frac{dT}{dt} = \frac{1}{C}(\eta \cdot A_{\mathrm{F}} \cdot (H(t) - T(t)) - k \cdot A_{\mathrm{w}} \cdot (T(t) - T_{\mathrm{zew}}(t)))$$

Legenda:

- T(t) – Temperatura wewnętrzna pomieszczenia w czasie (t) [°C].
- H(t) – Temperatura podłogi (czyli źródła ciepła) w czasie (t) [°C].
- $T_{zew}(t)$ – Temperatura zewnętrzna w czasie (t) [°C].
- η – Współczynnik efektywności przenikania ciepła z podłogi do powietrza w pokoju [W/m^2K].
- k – Współczynnik przenikania ciepła przez ściany budynku [W/m^2K].
- A_W – Całkowita powierzchnia ścian zewnętrznych pomieszczenia [m^2].
- A_F – Powierzchnia podłogi w pomieszczeniu [m^2].
- C – Pojemność cieplna pomieszczenia określająca ilość energii potrzebnej do podgrzania całego powietrza w pomieszczeniu o jeden stopień Celsjusza [J/K].

Wyjaśnienie powyższego równania:

Równanie to opisuje, jak szybko temperatura w pomieszczeniu zmienia się w odpowiedzi na działanie systemu ogrzewania podłogowego oraz wymianę ciepła z otoczeniem zewnętrznym. Współczynnik (η) mierzy, jak efektywnie ciepło jest przekazywane z podłogi do powietrza w pomieszczeniu, a współczynnik (k) odzwierciedla, jak szybko ciepło ucieka z pomieszczenia przez ściany zewnętrzne. Pojemność cieplna (C) mówi o tym, jak duża jest zdolność pomieszczenia do magazynowania ciepła.

<u>Równanie opisujące zmianę temperatury podłogi H(t) w czasie:</u>

$$\frac{dH}{dt} = \alpha(H_{\max} - H(t)) - \beta(H(t) - T(t))$$

Legenda:

- H_{max} – jest stałą reprezentującą maksymalną temperaturę, jaką może osiągnąć podłoga [°C].
- α – to współczynnik szybkości ogrzewania podłogi,
- β – to współczynnik szybkości chłodzenia podłogi,
- T(t) – to temperatura otoczenia wewnątrz budynku w czasie (t) [°C],
- H(t) – to temperatura podłogi w czasie (t) [°C].

W projekcie (module środowiska) przyjęto następujące wartości:

- Krok symulacji: 1 minuta
- Rozmiar pomieszczeń: 4m×4m×2.6m – Pojemność **V** = $4[\mathrm{m}] \cdot 4[\mathrm{m}] \cdot 2.6[\mathrm{m}] = \mathbf{41.6}[\mathrm{m}^3]$
- Każde pomieszczenie ma 2 ściany wewnętrzne i 2 zewnętrzne co daje powierzchnię ścian zewnętrznych: **A** = $2 \cdot 4[\mathrm{m}] \cdot 2.6[\mathrm{m}] = \mathbf{20.8}[\mathrm{m}^2]$
- Współczynnik efektywności przenikania ciepła **η** = $\mathbf{8.45}[\mathrm{W/m^2K}]$
- Pojemność cieplna pomieszczenia C = C_1 + C_2 = $\mathbf{651000}[\mathrm{J/K}]$
  Dla powietrza: (iloczyn objętości i gęstości powietrza oraz jego ciepła właściwego):
  C_1 = $\mathrm{V} \cdot \rho \cdot \mathrm{c} = 41.6[\mathrm{m}^3] \cdot 1.2[\mathrm{kg/m}] \cdot 1020[\mathrm{J/(kg \cdot K)}] = 50918.4[\mathrm{J/K}] \approx \mathbf{51000}[\mathrm{J/K}]$
  Wyposażenie pomieszczenia i samo pomieszczenie przyjęto C_2 = $\mathbf{600000}[\mathrm{J/K}]$
- Współczynnik przenikania ciepła przez ściany **k** = $\mathbf{0.8}[\mathrm{W/m^2K}]$
- Współczynnik szybkości ogrzewania podłogi **α** = $\mathbf{0.0025}[1/\mathrm{min}]$
- Współczynnik szybkości chłodzenia podłogi **β** = $\mathbf{0.005}[1/\mathrm{min}]$

Zmiana wartości współczynników wpłynie na jakość predykcji modelu, ale dopóki zmiany w środowisku będą przewidywalne dla 7 godzin historii wystarczy go dodatkowo przeszkolić.

Model A3C – Asynchronous Advantage Actor-Critic

Model A3C jest jednym z zaawansowanych algorytmów uczenia ze wzmocnieniem, który wykorzystuje zarówno podejście Actor-Critic, jak i Procesy Decyzyjne Markowa (MDP).

Najważniejsze informacje:

- Proces Decyzyjny Markova (MDP):

  W MDP interakcje agenta z otoczeniem są modelowane jako sekwencje stanów, akcji i nagród. Agent wybiera akcję na podstawie obserwowanego stanu środowiska, po czym środowisko reaguje, przenosząc agenta do nowego stanu i przyznając mu nagrodę.

  Celem agenta jest maksymalizacja sumy przyszłych nagród, co jest realizowane przez optymalizację polityki wyboru akcji.
- Actor i Critic:

  W modelu A3C wykorzystuje się dwie główne składowe: Actor'a (aktora), który decyduje o akcjach, i Critic'a (krytyka), który ocenia te decyzje.

  Actor generuje politykę działania, czyli rozkład prawdopodobieństwa wyboru każdej możliwej akcji w danym stanie.

  Critic ocenia, jak dobra jest akcja wybrana przez Actor'a w kontekście maksymalizacji przyszłych nagród. Critic szacuje wartość funkcji wartości stanu.
- Minimalizacja straty i nauka:

  Strata Actor'a jest wyznaczana na podstawie różnicy pomiędzy nagrodą, a przewidywaną wartością stanu (błąd TD, temporal difference error). Actor aktualizuje swoją politykę w kierunku zwiększania prawdopodobieństwa akcji, które prowadzą do stanów o wyższych nagrodach.

  Strata Critic'a wynika z różnicy między przewidywaną wartością, a rzeczywistą wartością uzyskaną po wykonaniu akcji. Critic aktualizuje swoje szacunki wartości stanów, aby były bardziej dokładne.

  Oba komponenty uczą się jednocześnie, gdzie Actor stara się maksymalizować przewidywane nagrody (korzystając z informacji zwrotnej od Critic'a), a Critic stara się dokładniej oceniać wartość stanów.
- Asynchroniczność:

  A3C wprowadza asynchroniczność poprzez równoczesne trenowanie wielu agentów w oddzielnych kopiach środowiska. Każdy agent działa niezależnie, co pozwala na efektywniejsze i szybsze eksplorowanie przestrzeni stanów.

  Asynchroniczność pomaga również w uniknięciu problemów związanych z korelacją sekwencji danych oraz z lokalnymi minimum podczas procesu uczenia.

Opis interakcji ze środowiskiem

Środowisko dostarcza dane co minutę, ale na potrzeby modelu przedstawiane, jako stan, są dane o temperaturach i czasie z 10 minutowymi przerwami.
Dokładnie działa to tak:
Dane zapisywane są w tablicy (macierzy) co minutę – jeden wektor zawierający aktualny stan pomieszczenia. Następnie dobierane są odpowiednie wektory, które przekazywane są do modelu jako stan. Oznacza to:
42 wektory odpowiednio wektor: {410, 400, ..., 20, 10, 0} w następnym kroku to wektory o 1 większe {411, 401, ..., 21, 11, 1} itd. brakujące wektory, na początku działania aplikacji, wypełniane są takimi wartościami jak początkowe. Dzięki temu każdy kolejny krok daje inny, choć zbliżony do poprzedniego, stan na wejściu do sieci neuronowej. Sieć próbuje dobrać akcję (zamknięcie / otwarcie) tak aby zminimalizować zdefiniowaną w środowisku karę. Składają się na nią następujące czynniki: różnice temperatur między zadaną, a aktualną oraz między maksymalną dla podłogi i jej aktualną temperaturą (o ile aktualna jest wyższa).

Każdy Agent działa w osobnym procesie na indywidualnym środowisku i jednocześnie obsługuje wiele pokoi, tak aby w każdej epoce zebrać możliwie jak najwięcej danych do aktualizacji głównego modelu.

Na koniec epoki dane ze wszystkich procesów Agentów są zbierane oraz mieszane. Tak przygotowane dane służą do uczenia modelu głównego, którego wagi są następnie propagowane do modeli Agentów i cały proces powtarza się przez zadaną liczbę epok.

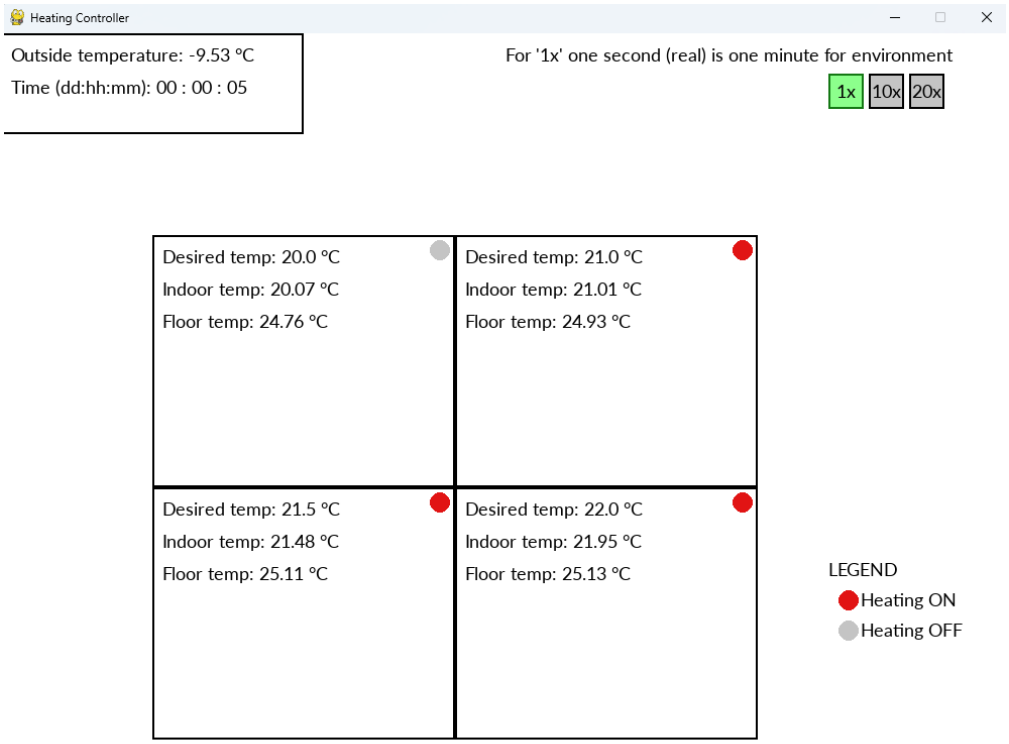
Analiza Danych i Wizualizacja

Dla ‘trybu cichego’ wykorzystano biblioteki Pandas i Matplotlib. Dane są analizowane i wizualizowane w formie wykresów dla pojedynczego pomieszczenia z góry ustaloną zadaną temperaturą, co pozwala na ocenę efektywności modelu sterującego ogrzewaniem oraz strategii zarządzania temperaturą.

Wykresy na drugiej i trzeciej stronie zostały stworzone właśnie za pomocą tego modułu. Moduł tworzy również dane tabelaryczne zapisywane do plików *.csv oraz wypisuje w konsoli informacje dotyczące odchyleń standardowych dla temperatury wewnątrz pomieszczenia dla każdego z użytych modeli sterujących.

Interaktywna Wizualizacja w Pygame

Dodatkowy moduł – symulator – stworzony z pomocą Pygame zapewnia interaktywną wizualizację pracy modelu w czasie rzeczywistym z możliwością przyspieszenia upływu czasu. Umożliwia to obserwowanie efektów działania systemu w przystępnej formie wizualnej. Czerwona kropka w pokoju oznacza grzanie, szara brak ogrzewania.



Technologia

- Python
- TensorFlow
- Numpy
- Pandas
- Matplotlib
- Pygame

Autor

Daniel Poloczek

daniel.poloczek.it@gmail.com

https://www.linkedin.com/in/daniel-poloczek-785161185/