Semestrální práce z UPG

David Konečný

22. prosince 2024

Elektrostatické pole

1 Implementované řešení

Projekt vizualizace elektrostatického pole umožňuje zobrazit náboje a potenciál elektrostatického pole v reálném čase. Na základě parametrů definovaných uživatelem se pole aktualizuje a vykresluje. Aplikace využívá strukturu Vector2D pro výpočty s dvourozměrnými vektory a jejich vizualizaci ve scénáři elektrostatického pole.

Řešení je strukturováno do několika tříd:

- `Program`: Hlavní vstupní bod aplikace, který přijímá volitelný parametr scénáře a předává ho dále do aplikace.
- `MainForm`: Hlavní okno aplikace (formulář), které spravuje a inicializuje scénáře a zajišťuje přístup ke grafickému panelu.
- `DrawingPanel`: Grafický panel, který vykresluje náboje a vektory elektrostatického pole na základě výpočtů.
- `Vector2D`: Struktura pro práci s dvourozměrnými vektory, která je používána při výpočtech a kreslení směrů siločar.
- `ProbeChart`: Okno zobrazení grafu závislosti intenzity elektrického pole na čase.
- `Charge`: Třída reprezentující elektrický náboj s informacemi o pozici a velikosti.
- `Vector2DJsonConvertor`: Třída pro převod načtených hodnot ze souboru (JSON).
- `ChargeEditor`: Formulářové okno pro zadání hodnot `position` a `magnitude` nového náboje.

2 Dekompozice do tříd a balíčků

Projekt je dekomponován do následujících tříd a balíčků:

- `Program`: Obsahuje hlavní metodu, která spouští aplikaci a volitelně přijímá argument pro výběr scénáře.
- `MainForm`: Hlavní GUI formulář, který volá metody k inicializaci a správě scénářů, včetně komunikace s panelem pro kreslení.

- `DrawingPanel`: Klíčová třída pro grafické vykreslení pole. Obsahuje metody pro vykreslování vektorů, šipek a nábojů na základě dat získaných ze scénáře.
- `Vector2D`: Struktura, která umožňuje práci s dvourozměrnými vektory. Zajišťuje operace jako sčítání, odčítání, násobení, dělení skalárem, výpočet velikosti a normalizaci vektoru.
- `Charge`: Třída zajišťující reprezentaci bodových nábojů s vlastnostmi `position` a `magnitude`.
- `ProbeChart`: Třída zajišťující vykreslování grafu intenzity elektrického pole v čase.
- `Vector2DJsonConvertor`: Override metody `read()` a metody `write()` třídy JsonConverter, pro správné načtení dat z JSON souboru.
- `ChargeEditor`: Implementuje třídu, která dědí od třídy `Forms` a na zavolání zobrazí okno pro zadání hodnot nového náboje.

3 Zavedená omezení a zjednodušení

- Scénáře jsou pevně definované v kódu, a uživatel nemá možnost zadat vlastní scénáře přímo přes GUI.
- Pro vykreslování pole jsou vektory normalizovány, což může vést k přibližné reprezentaci intenzity.
- Simulace nepodporuje komplexnější dynamické interakce, jako je pohyb nábojů nebo reaktivní změny konfigurace.

4 Klíčové algoritmy

4.1 Výpočet elektrického pole

Algoritmus pro výpočet intenzity elektrického pole je implementován v metodě `CalcElectricFieldAt()` třídy `DrawingPanel`. Intenzita v daném bodě je součtem příspěvků všech nábojů podle Coulombova zákona:

- Každý náboj je reprezentován jako bodový zdroj pole s danou pozicí a velikostí.
- Vzdálenost a směr mezi nábojem a bodem výpočtu se určuje pomocí vektorových operací.
- Výsledné vektory intenzity jsou sčítány do celkového výsledku.

4.2 Kreslení intenzity a vektorů

Pro zobrazení intenzity elektrického pole je využita metoda `DrawIntensityMap()` třídy `DrawingPanel`. Tato metoda předpočítává intenzitu pole pro jednotlivé body v mřížce a barvami zobrazuje jejich hodnoty. Vektory pole jsou kresleny pomocí šipek reprezentujících směr a velikost.

4.3 Graf intenzity pole v čase

Třída `ProbeChart` obsahuje metodu `UpdateChart()`, která průběžně aktualizuje hodnoty na základě intenzity pole v místě druhé sondy. Intenzita je vzorkována každých 50 ms a přidávána do grafu.

4.4 JSON convertor

Ve třídě `Vector2DJsonConvertor` jsou přepsané metody `read() & write()`. Data ze vstupního souboru vypadají například.:

```
[
{"position": {"X": -1, "Y": 0}, "magnitude": 1},
{"position": {"X": 1, "Y": 0}, "magnitude": -2},
{"position": {"X": 0, "Y": 1}, "magnitude": 0.5}
]
```

Hledá se `propertyName` "X"nebo "Y", poté se přiřadí hodnoty a vytvoří se instance `Vector2D`, která představuje pozici náboje.

5 Spuštění

1. Projekt lze sestavit pomocí C# prostředí (např. Visual Studio) nebo pomocí souboru build.
2. Spuštění aplikace lze provést s volitelným parametrem scénáře z příkazové řádky nebo pomocí souboru run.
3. **Příklad spuštění**: `ElectricFieldVis.exe 1` pro spuštění scénáře 1. Nebo `ElectricFieldVis.exe C:\cesta\k\json\souboru` pro načítání dat ze souboru.

6 Ovládání aplikace

- **Myš:**
  - Kliknutím a tažením lze přemístit náboje.

  - Kliknutí levým tlačítkem do prostoru přidá druhou sondu, jejíž data jsou zobrazena v grafu.
  - Rolování kolečkem myši změní velikost vybraného náboje.

- **Parametry příkazové řádky:**
  - Argument `-g` nastaví velikost mřížky (např. `-g10x10`).
  - Číselný argument určuje scénář (např. `2` spustí scénář 2).

7 Nedostatky a možná rozšíření

7.1 Nedostatky

- Aplikace neřeší kolize nebo pohyb nábojů, což omezuje možnosti simulace.
- Vykreslování vektorů je omezeno na normalizovanou velikost, což může zkreslit intenzitu pole.
- Graf intenzity pole neumožňuje uživatelskou úpravu nebo ukládání dat.

7.2 Možná rozšíření

- Přidání dynamických scénářů, kde se náboje pohybují nebo mění své vlastnosti.
- Umožnění vytváření vlastních scénářů prostřednictvím GUI.
- Zobrazení nenormalizovaných vektorů intenzity, které lépe odrážejí skutečné rozdíly v intenzitě pole.
- Rozšíření grafického rozhraní o možnosti ukládání a exportu dat.