Dokumentace k programu VISU - PYTH

Program VISU je určen pro visualizaci výstupu z programu pro 2D simulaci hydrodynamiky pomocí lagrangeovských cell-centered numerických metod.

Operační instrukce

Program je koncipován jako jediná rutina, na jejíž vstup se mimo jiné vkládají textové soubory obdržené ze simulačního programu. Tyto soubory potřebné pro visualizaci jsou celkem čtyři:

- `input_cellsid.dat` Jedná se o datový soubor obsahující seznam ID buněk, jejich identifikátor hranice, počet jejich uzlů a počet napojených buněk.
- `input_cnconn.dat` Datový soubor obsahující ID buněk, seznam uzlů náležících buňce a seznam příslušných vedlejších buněk.
- `mesh_xxx.dat` Datový soubor obsahující souřadnice uzlů sítě, uspořádány dle ID uzlu.
- `vals_xxx.dat` Datový soubor obsahující hodnoty veličin uložených v centru buněk (hustota, tlak, vnitřní energie) uspořádaných dle ID buněk. Užívá se pro vykreslení konturovaného grafu.

  Trojčíslí `xxx` označuje pořadí výstupního souboru ve smyslu výpočtu v simulačním programu, konkrétně soubor s označením `000` představuje počáteční stav simulace a vyšší čísla jsou stavy v čase $t > 0$.

  Poslední parametr nutný pro spuštění je veličina k barevnému vykreslení výstupu. Jedná se o řetězce `'crho'` pro hustotu, `'cEni'` pro vnitřní energii a `'cp'` pro tlak unvitř buněk.

Pro spuštění je tedy celkově nutné zavolat rutinu `draw` s následujícími parametry:

```
draw(cesta k mesh_xxx.dat,
     cesta k input_cellsid.dat,
     cesta k input_cnconn.dat,
     cesta k vals_xxx.dat,
     veličina k vykreslení)
```

Organizace programu

Vnitřní datová struktura je v jádru realizována dvěma třídami, třídou `Node` a třídou `Cell`. Třída `Node` má atributy ID a souřadnice x,y. Třída `Cell` obsahuje atributy ID, počet uzlů, seznam vnitřních uzlů a seznam hodnot v cell-centru.

Při inicializaci se z příslušných souborů na vstupu importují potřebná data a vytvoří se seznamy objektů `Node` se všemi atributy a `Cell` s atributy ID a počet uzlů. Seznam uzlů má název `nodes` a seznam buněk `cells`. Indexace těchto seznamů je přirozená dle ID, tj. k-tý prvek seznamu odpovídá uzlu nebo buňce s ID o hodnotě k. Samotný import z datových souborů je realisován skrze knihovnu `pandas`. Zbylé atributy objektů třídy `Cell` jsou dodány pomocí cyklů, přičemž podoba seznamu vnitřních uzlů je přirozeně závislá na jejich počtu, což je ošetřeno `if` podmínkou kladenou na počet uzlů. Tato procedura je implementována pouze pro sítě sestávající se ze troj- až šestiúhelníků. Důvod je ten, že při simulaci se sítě s polygony s počtem vrcholů větším než 6 prakticky nepoužívají. Pokud by k tomu však v budoucnu došlo, není problém tuto podmínku rozšířit. Hodnoty veličin v cell-centru se taktéž přidávají v cyklu, přičemž finálním atributem v k-tém prvku seznamu buněk `cells` je indexovaný seznam v následujícím tvaru:

```
cells[k].ctr_vals = [x-souřadnice,
                     y-souřadnice,
                     x-souřadnice rychlosti,
                     y-souřadnice rychlosti,
                     hustota,
                     tlak,
                     vnitřní energie]
```

Po vytvoření datové struktury a importu dat je nutné připravit colormap pro vykreslení hodnot uvnitř buněk. Toto se provádí skrz knihovnu `matplotlib` pomocí metody `matplotlib.colors.Normalize`, která převede hodnoty v libovolném intervalu $< a, b >$ do intervalu $< 0, 1 >$. V této fázi se v závislosti na vstupním parametru veličiny k vykreslení nalezne minimum a maximum seznamu hodnot zvolené veličiny ve všech buňkách, přičemž poté se provede samotná normalizace a normalizovaný seznam se vloží dle ID zpět do buněk.

Samotné vykreslení probíhá následovně. Na výstupu z programu se vytvoří tři obrázky. Prvním je graf s vykreslenými uzly sítě.

Druhým obrázkem je contour plot s vyobrazenou sítí a obarvením buněk dle zvolené hodnoty v jejich středu. Toto je implementováno s pomocí balíčku `PatchCollection` uvnitř knihovny `matplotlib`, který umožňuje jednoduše kreslit polygony. Pro využití příslušných metod pro zobrazení polygonů je nutné souřadnice, a tedy i ID uzlů v buňce podávat proti směru hodinových ručiček, což je ošetřeno na straně výstupu ze simulačního programu. Polygony se kreslí v cyklu, do nějž je třeba vložit seznamy x-ových a y-ových souřadnic ve zmíněném uspořádání a poskytnout hodnotu z colormapu pro obarvení polygonu. Celá síť se tímto způsobem kreslí buňka po buňce.

Posledním obrázkem na výstupu je samotná síť.

Poznámky

- Uvnitř programu je navíc implementována funkce `connect_nodes`, která vytváří čáru mezi dvěma zvolenými uzly. Tato funkce není momentálně nijak využita, ale v některých případech se může hodit pro účely testování simulačního programu.

- Samotný program je psaný tak, aby bylo možné přidávat dodatečné funkcionality v závislosti na vývoji simulačního programu, zejména o užité typy sítí. Každý typ sítě generovaný simulačním programem je co se týká datové struktury stejný, takže jediné místo, kde by potenciálně bylo nutné provádět modifikace, je rutina pro plnění buněk seznamem jejich vnitřních uzlů, přesahuje-li jejich počet 6. Jak jsem však zmínil dříve, nejpoužívanější jsou trojúhelníkové a čtyřúhelníkové sítě, což je v tomto programu již ve vší obecnosti pokryto.

- Při importu ze souboru s konektivitou sítě je možné si všimnout, že v každém jeho řádku je různý počet hodnot. Jelikož je v programu potřeba z tohoto datasetu jen ID buňky a seznam uzlů v buňce, tato skutečnost nevadí. Pokud by však bylo zapotřebí užít i seznamu vedlejších buněk, není možné tato data použít přímo ze setu `data2`. Místo toho by bylo nutné sestavit parser a data neimportovat pomocí knihovny `pandas`. Jelikož se však formát vstupních souborů v průběhu vývoje simulačního programu několikrát měnil, ponechal jsem toto místo neošetřené, jelikož není podstatné pro funkcionalitu programu.