Sistemi Multimediali e Laboratorio

Roberto Ranon

Esercitazione n.3 del 13/03/2018: particelle e animazioni con leggi fisiche - parte 2

Nota: *per svolgere questi esercizi, dovete aver scaricato e installato l'ultima versione di Processing (attualmente, la 3.3, ma qualsiasi versione dalla 2.x dovrebbe andare bene). Se state lavorando su una delle macchine del laboratorio, dovreste trovare Processing tra i programmi disponibili in ambiente Windows. Se ci sono problemi, avvisatemi durante la lezione o con un mail a roberto.ranon@uniud.it.*

Le slide che presentano brevemente i concetti e i comandi necessari sono disponibili all'indirizzo ***https://users.dimi.uniud.it/~roberto.ranon/slides/processing-intro.pdf*** *(o sul sito* ***elearning.uniud.it****). Per maggiori informazioni, consultate la guida di riferimento di Processing: basta scrivere nell'editor l'istruzione, selezionarla, cliccare su di essa con il tasto destro, e scegliere "Find in Reference".*

Ci aspettiamo che, al termine del laboratorio, abbiate completato almeno gli esercizi fino al n. 5. Se non è così, cercate di completarli a casa.

1. Partite dal codice di esempio (**Es3-startingCode.zip** sul sito e-learning). Il codice definisce una classe `Particle`, con vettori 2D per rappresentarne posizione, velocità e accelerazione, e inoltre usa la *seconda legge di Newton* per calcolare l'accelerazione A in base ad una forza impressa alla particella (un vettore 2D *F*) e alla massa della particella (uno scalare *m*):

   $$A = F / m$$

   In particolare, il metodo `ApplyForce(PVector force)` implementa la relazione appena descritta, calcolando l'accelerazione in base alla forza impressa, e sommandola all'accelerazione corrente. Inoltre, osservate nel codice di partenza:

   - che nel metodo `Update`, dopo aver aggiornato la velocità con l'accelerazione, e aver calcolato la nuova posizione, poniamo l'accelerazione a zero con l'istruzione `acceleration.mult(0)`. Questo è necessario, altrimenti l'accelerazione andrebbe a sommarsi a quella che avevamo nel frame precedente;
   - che nel metodo `CheckBorders()`, introduciamo delle istruzioni per evitare che la nostra particella oscilli troppo velocemente o esca completamente dallo schermo
   - che, a seconda dei casi, usiamo le operazioni tra vettori in due modi diversi, ad esempio:
     - `position.add(velocity);` modifica il valore di position
     - `PVector.div(force,mass);` divide `force` per `mass` senza modificare `force`, ma restituendo il risultato
     - a seconda di ciò che vogliamo, dovremo usare la forma adatta ad evitare side-effect indesiderati.

   Dopo aver esaminato attentamente il codice di partenza, modificatelo in modo che la particella sia sottoposta a due forze: vento proveniente da sinistra (con modulo uguale a 0.01) e gravità (con modulo uguale a 0.01).

2. Modificate l'esercizio precedente in modo che:

   - il diametro della particella sia proporzionale alla sua massa (scegliete voi il rapporto tra i due)

   - vengano generate 20 particelle, con massa random compresa tra 0.1 e 4, tutte con posizione iniziale pari a (0,0). Impostate, inoltre, nel metodo display, un valore di alpha per il riempimento delle particelle pari a 128 [1].

3. In fisica l'**attrito** [2] è una forza che si oppone allo scivolamento o rotazione di un corpo su una superficie, in pratica, rallentandolo. Una formula un po' semplificata per calcolare l'attrito in movimento è:

   $$\text{Friction} = -1 \bullet c \bullet \underline{v}$$

   dove $\underline{v}$ è la direzione della velocità (cioè, un vettore di lunghezza unitaria che ha la stessa direzione del vettore velocità corrente), e *c* è un coefficiente di attrito che possiamo impostare a piacere per aumentare o diminuire l'effetto dell'attrito.

   Modificate l'esercizio precedente, in modo che le particelle siano sottoposte, oltre alla gravità e al vento, anche ad un attrito con *c=0.05* (provate anche a variare il valore di c per vedere cosa succede).

4. L'attrazione gravitazionale tra due oggetti è governata dalla formula $\mathbf{F}=(Gm_1m_2)/r^2 \times \underline{\mathbf{r}}$, dove:

   - G e' la costante gravitazionale universale, un numero reale. Nel nostro caso, non è importante che abbia il valore di riferimento (6.67428×10^{-11}), ma un valore adatto ai nostri scopi (provate con il valore `1`, all'inizio).

   - m_1 ed m_2 sono le masse degli oggetti coinvolti

   - r è la distanza tra i due oggetti (ad esempio, i due centri). La distanza può essere calcolata con il metodo `dist()` di Processing (https://processing.org/reference/dist_.html), oppure, ancora meglio, sottraendo il vettore posizione di uno dal vettore posizione dell'altro, e calcolando il modulo del vettore risultante (metodo `mag()`, https://processing.org/reference/PVector_mag_.html).

   - $\underline{\mathbf{r}}$ è un vettore di lunghezza unitaria che punta dall'oggetto 1 all'oggetto 2 (se 2 attrae 1, o viceversa, se 1 attrae 2). Si può calcolare sottraendo la posizione dell'oggetto 1 dall'oggetto 2, e normalizzando il risultato.

Scrivete uno sketch Processing che visualizzi:

- una particella, ottenuta modificando la classe Particle, il cui moto è governato dalla forza di attrazione gravitazionale, in modo che la particella sia attratta da un oggetto *attrattore*.

---

[1] In realtà, la forza di gravità dovrebbe essere scalata rispetto alla massa; in questo modo, si ottiene un comportamento più realistico. Una volta terminato l'esercizio, modificatelo in questo senso, e verificate che cosa succede

[2] https://it.wikipedia.org/wiki/Attrito

- un oggetto “attrattore", disegnato tramite un’ellisse, dotato di una certa massa e immobile, al centro dello schermo. Suggeriamo di creare una classe per l’oggetto attrattore, con un metodo che calcoli la forza con cui questo attrae l’oggetto passato come argomento.

5. Modificate l’esercizio precedente in modo che vengano create 20 particelle, con massa random compresa tra 0.1 e 4, in posizione casuale, e tutte attratte dall’attrattore al centro.

6. Create un sistema di 50 particelle, ognuna con massa pari a `1` e accelerazione iniziale casuale. Ogni volta che avviene una collisione tra due particelle, esse si devono respingere. L’esercizio vi chiede di risolvere due problemi:

   - rilevare la collisione tra due particelle. Per fare questo, basta controllare se la distanza tra le due particelle è minore o uguale alla somma dei due raggi. In caso positivo, è avvenuta una collisione.

   - calcolare la forza di repulsione. Ad esempio, per due particelle `p1` e `p2`, per `p1` la forza deve essere l’opposto del vettore che collega il centro di `p1` al centro di `p2`, per quanto riguarda la direzione, e il modulo può essere impostato ad un valore a piacere; per `p2`, vale il medesimo ragionamento.

   Dovete poi fare in modo che la rilevazione della collisione e l’eventuale applicazione della forza di repulsione avvenga tra ogni coppia di particelle.