Animation

Schreiben Sie ein MATLAB-`Skript` `animation.m`, das folgende Aufgaben erfüllt:

Es soll die Bewegung von Punkten entlang Kurven in einer Animation darstellt werden. Ein Punkt bewegt sich dabei entlang einer elliptischen Bahn, die unter Umständen im Laufe der Zeit eine Verkleinerung der Achsen erfährt.

1. Diese Bahn wird beschrieben durch

$$x_1(t) = a_1 cos(2\pi\nu_1 t)e^{-\kappa_1 t} \ , \tag{1}$$
$$y_1(t) = b_1 sin(2\pi\nu_1 t)e^{-\kappa_1 t} \ . \tag{2}$$

   Dieser Punkt wird von einem zweiten Punkt umrundet, wobei seine Bahn durch

$$x_2(t) = x_1(t) + a_2 cos(2\pi\nu_2 t)e^{-\kappa_2 t} \ , \tag{3}$$
$$y_2(t) = y_1(t) + b_2 sin(2\pi\nu_2 t)e^{-\kappa_2 t} \tag{4}$$

   gegeben ist.

2. Der Zeilen- Zeitvektor t soll dabei von 0 bis t_{end} laufen. Verwenden Sie zum testen eine sinnvolle Kombination von Frequenzen ν, Achsen a und b, Dämpfungsfaktoren κ und Zeitwerten. Der erste Punkt soll dabei ein paar Umläufe machen, während in der zweite Punkt pro Umlauf den ersten mehrfach umkreist.

3. Später, bei der Abgabe, werden diese Variablen
   `a_1, a_2, b_1, b_2, kappa_1, kappa_2, nu_1, nu_2, t`
   von den Automatischen Tests vorgegeben, und sollten im Script nicht überschrieben werden!

Verwenden Sie dabei zur Erstellung der Graphik die Befehle `figure`, `axes`, `line`. Diese (alle) Graphikbefehle haben als Rückgabewert einen sogenannten Graphik-Handle, der einen weiteren Zugriff auf diese Graphikobjekte ermöglicht:

```
fh = figure;
clf reset;    % loescht alle Children, Eigenschaften
set(fh,'Color',[1,1,0.8]); % Setzt die Hintergrundfarbe des Plotfensters
set(fh,.........)
```

Damit kann man alle Eigenschaften von Graphikobjekten verändern. Dazu kann man sich im `helpbrowser` die Seite "Handle Graphics Property Browser" ansehen, die für alle Graphikobjekte Eigenschaften und mögliche Werte angibt. Eine Einführung zu diesem Konzept gibt es im Kapitel 15 .

Das gleiche Konzept wird zur Animation von Graphiken verwendet. Dabei tauscht man in einer Schleife (`while`) z.B. die Daten einer Linie (eines Punktes) aus:

```
lh = line(x,y,'LineStyle','none');
set(lh,'Marker','o')
% MarkerSize, MarkerEdgeColor, MarkerFaceColor, ....
for k = [1:numel(t)] % k über t laufen lassen
  ti = t(k);    % den k-ten Wert aus vector t holen
  x = ....
  y = ....
  set(lh,'XData',x,'YData',y);
  drawnow;
  pause( .......... )
end
```

Die Animation erfolgt also durch Austauschen der x- und y-Werte. Der Befehl `drawnow` stellt sicher, dass zu diesem Zeitpunkt ein Update der graphischen Ausgabe (Schirm) stattfindet. MATLAB merkt sich ansonsten Graphikbefehle in einer Art Pipeline und optimiert die Umsetzung.

1. Erstellen sie 2 line Objekte. Der Handle des ersten line soll `lh_1` sein. Das line Objekt für den zweiten Punkt (das ist jener, welcher um den ersten Punkt rotiert) soll den Handle `lh_2` haben. Erstellen Sie zuerst `lh_1`, dann `lh_2`.

2. Die Punkte des jeweiligen Line Objekts sollen nicht mit Geraden verbunden werden. (Diese Forderung macht erst Sinn, wenn man den Letzten Absatz liest. Im moment bestehen die jeweiligen Line-Objekte noch aus jeweils einen Punkt.)

3. Bei den Line Properties soll als Marker 'o' genommen werden.

4. Der Marker des ersten Punktes (`x_1,y_1`) soll rot aufgefüllt sein, der zweite (`x_2,y_2`) grün. Die restlichen Markereigenschaften werden nicht überprüft, und können entsprechend den eigenen ästhetischen Ansprüchen beliebig gesetzt werden.

5. Die Eigenschaften `XLim` bzw. `YLim` der Achsenlimits sollen gleich am Anfang in entsprechender Größe eingestellt werden, damit sich diese nicht bei der Bewegung mitändern. Gehen Sie davon aus, dass der Zeitvektor mit 0 anfängt ($t_1 = 0$, und monoton ansteigend). Dann ist man mit

$$x_{max} = |a_1| + |a_2| \tag{5}$$

   auf der sicheren Seite. Die anderen Grenzen sind analog zu berechnen.

6. Mit dem `pause` Befehl können Sie die Animation verlangsammen.

Die Animation sollte nun Funktionieren.

1. Wenn dieser Teil funktioniert, ergänzen Sie die Ausgabe um "Schweife" beschränkter Länge, die jeder Punkt nach sich zieht. Diese stellen quasi eine Spur dar. Dafür führt man die Linien so ein, dass sie nun nicht mehr nur aus einem Punkt sondern aus einer Reihe von Punkten bestehen.

2. Dies kann man dadurch erreichen, dass man an die entsprechenden Vektoren bei jedem Zeitschritt vorne den neuen Punkt anfügt.

3. Wenn die Vektoren die gewünschte Länge überschreiten, schneidet man den hinteren Bereich ab.

   Die Animimation erfolgt wie vorher unter Verwendung von `XData` und `YData`.

4. Verwenden Sie als Anzahl der Punkte pro Line- Objekt incl. "Schweif" `L = 5`.

Hinweis:

Bevor Sie das Script abgeben, sollten sie die Initialisierungen von
`a_1, a_2, b_1, b_2, kappa_1, kappa_2, nu_1, nu_2, t`
auskommentieren.

Hinweis:

Um Animationen oder auch videos zu erstellen gibt es eine elegantere Möglichkeit: siehe `movie` und `avifile`. Dabei wird allerdings extrem viel Arbeitsspeicher benötigt. Es könnte durchaus 1MB pro frame gebraucht werden. Die automatische Überprüfung ist deshalb praktisch nicht realisierbar.

Gesucht: Script `animation.m`

Anschauungsbeispiel: