前回お話しした加速度の話は「等加速度直線運動」と言って、加速度が一定で速度がドゥンドゥン増えていくものです。

初速度がゼロの場合は、加速度を a、その時の経過時間を t とすると、一定時間後の速度 v は

$$v = at$$

となります。

ここでは初速度ありの話をしていきます。
初速度というのは加速が始まった時に既にかかっている速度のことです。

例えばスケートとかで、一定の速度で滑っている時、ここには加速度がかかっていません。(速度の変更が行われていない時点で加速度はかかっていないといえる。)

ε ＝(´・ω・)
一定速度

ここで後ろから押される(力がかかる)と加速度がかかります。

(`・ω・)」」**ドン**ε ε ＝(´・ω・)

ところで力学でいうところの「力」というのは Force の F で表されるのですが、物理の定義として

$$F=ma$$

となっています。しかし、とりあえずはこれは置いておいて「**力がかかるってことは加速度がかかる**」という事だということを覚えておいて下さい。

若干話がそれましたが、一定速度で走っている最中に加速度がかかることがあります。この時の速度を「初速度」と言います。
初速度つきの速度と加速度の関係は

$$v=v_0+at$$

となります。なお v_0 ってのは初速度のことで、力学では初速度を V_0 と表記することが多いです。

ではここで質問です。

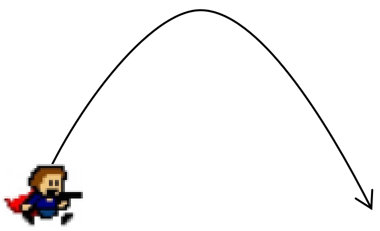
前回、移動距離と加速度の関係を話しました。
三角形の面積の考え方より

$$d=\frac{1}{2}at^2$$

が導き出せてました。では、ここに初速度が絡んでくるとどういう式になるでしょうか?以下に書いてみてください。ここも面積で考えればわかるはずです。

ここまではなんてことはないですね。
では次に次元を増やします。みんな大好き二次元です。

マリオ的なジャンプを考えてみましょう。

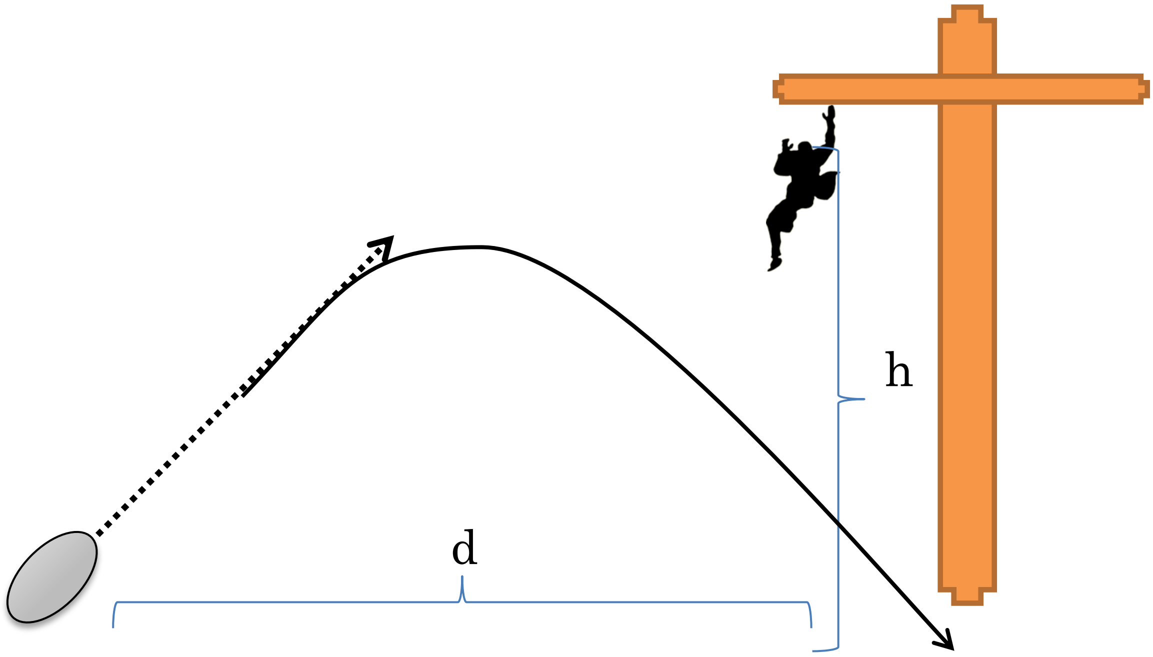


真上にジャンプしてそのまま降りるよりも、斜め方向にジャンプすることのほうが多いはずです。ということはこういう問題を解く必要がでてくるでしょう。

キッド君がx方向にv_xで進みつつジャンプしました。ジャンプの瞬間から着地点までの距離を求めなさい。なお、ジャンプの瞬間のy方向の上方向の速度をv_{y0}とし、重力加速度(等加速)をgとします。
ジャンプから着地までの時間tをv_{y0},gを使って表しなさい

着地点までの距離dをv_x,v_{y0},gを使って表しなさい

応用です。カプコソの入社試験でも出たという問題の改変です。
モンキーハンティングって問題です。
とある木にサルが捉まっております。ハンターはサルを撃ち殺したいので石を投げます。
石を投げるのに気合を入れるため、ハンターは「キェエエエエエエエ!!!」と叫びます。
サルは石を投げた瞬間に木から手を離して落っこちます。



ここで問題です。

ただし、ハンターからサルまでのX距離をdとし、ハンターとの高低差をhとします。

なお、重力加速度をgとし、石の発射速度(速度ベクトル長)をvとします。x方向とy方向を合わせてv

(1) ハンターは自分の位置からどれくらいの高さを狙えばうまく猿に当たりますか?gとかhとか使って表しなさい。

(2) 石とサルが命中した地点のX方向距離を求めなさい。

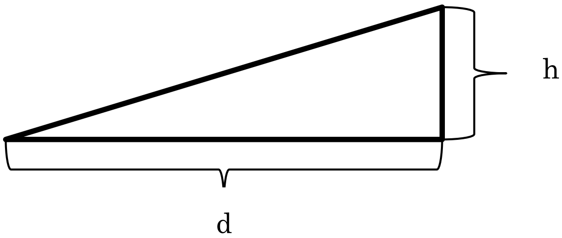
(3) 石を投擲してから、(1)の条件でサルに石が命中した瞬間までの時間tをvとかdを使って表しなさい。
(4) 石とサルがヒットした瞬間の、ハンターとの高低差をgとかvとかtを使って表しなさい。

…さて、前回までの『速度加速度』の話ですが、思ったよりも苦戦したのではないでしょうか？
特にモンキーハンティングには納得行ってない人も多いのかもしれません。なんか詭弁ぽく感じた人もいるでしょう。というわけで今一度運動方程式を見なおしてみましょう。

$$石y座標 = v_{y0}t - \frac{1}{2}gt^2$$

$$サルy座標 = h - \frac{1}{2}gt^2$$

こんな感じでしたね？つまり$v_{y0}t = h$ ですね。例えば前回の問題(3)を解くならば



こうなる角度に向かって投げるべきですよね？つまり、

$$v_{y0} = v_0 \times \frac{h}{\sqrt{h^2 + d^2}}$$

こうですよね？となると…

$$t = \frac{\sqrt{h^2 + d^2}}{v_0 h} \times h = \frac{\sqrt{h^2 + d^2}}{v_0}$$

というわけです。
…まぁ、ここまで来ても納得いかない人も多いでしょうからヴィジュアルに訴えかけてみましょうか。

チョットその前に軽く言っておきたいことがあります。あの$v_0t + \frac{1}{2}at^2$ という運動方程式は**加速度を二回積分**した値となっているのです。逆に言うと移動量を二回微分したものが加速度になっています。
ですから加速度とは

$$a = \frac{d^2x}{dy^2}$$

なんて書けちゃったりするんですが、そういう話はまた後々でいたしましょう。まだ早いですよ。

さて引き伸ばしてしまいましたがプログラムです。ゲームプログラムの場合はフレーム単位で時間が進むので t の部分が「秒」でなく「フレーム」となります。

で、先ほど積分の話しをちょろっとしたのには理由があって、ゲームプログラムの場合例えば

$$d = v_0 t + \frac{1}{2} a t^2$$

という数式は、ゲームループの中での

```
v=v0;//初速度を設定
ゲームループ{
	v+=a;//加速
	d+=v;//進む
}
```

に相当するわけです。あまりにシンプルになってしまうので妥当性があるのかどうかを疑問視する人もいるかもしれませんが、上のように「僅かに進んだ部分から累積して計算し、答えを導き出す」方法を Verlet 積分といいます。
興味がある人は
https://www1.doshisha.ac.jp/~bukka/lecture/jikkenn/simulation/verlet-3.html
でも見といて下さい。たぶん微積分からん人はよく分からんと思いますが、真ん中くらいに
「x(t)と x(t-h)がわかっていれば x(t+h)が計算できる」とありますが、この性質を用いてるのがゲームにおける「移動」なわけね。

まぁ、難しいことは置いておいて、ゲームのプログラムは学校でやる物理の計算よりもシンプルにできるってことだ。
¥¥132sv¥gakuseigamero¥rkawano¥数学¥0CPhysics.lzh
を解凍して、
Player.cpp を見てくれ。
そこの 47 行目くらいから①～⑦のコメントが有ると思う。そこに移動(加速度あり)の処理や、落下処理などのコードを書いてくれ。

書けましたか？色々と試してみればわかりますが、上のようなシンプルな加速度計算でキチンと等速直線運動、落下運動、ジャンプ運動などが可能であることがわかったと思う。

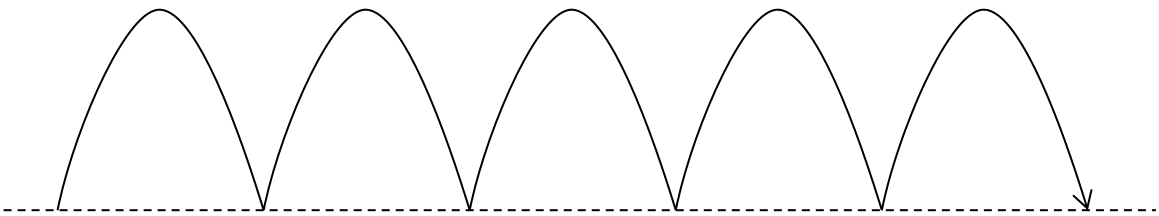
実はここに書いてある On〜って関数はメインループから毎フレーム呼び出される関数である Update から呼び出されているため、これらのことが可能である。

それではこのコードを参考に、前回みなさんが苦戦した「モンキーハンティング」をプログラムでシミュレートしてみてくれ。

余裕のある人は「バウンド」も実装してみて、ワンバン、ツーバウンドでもきちんと衝突する事を確認してくれたまえ。

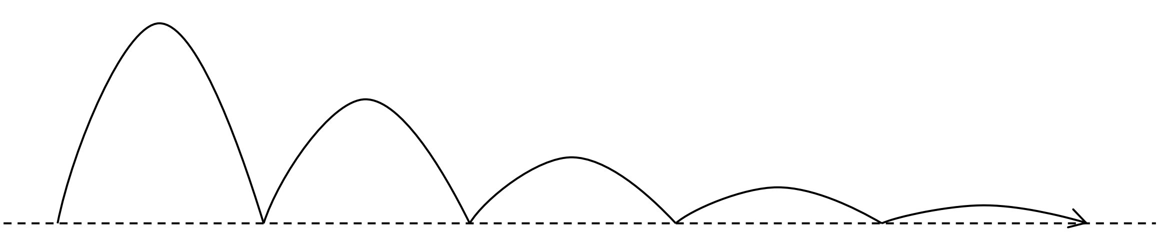
ちなみにバウンドとは簡単に言うと「はねかえり」である。

ヒントを言っておくと「はねかえり係数」が1なら跳ね返る直前のスピードが反転し、係数が0なら跳ね返らない。



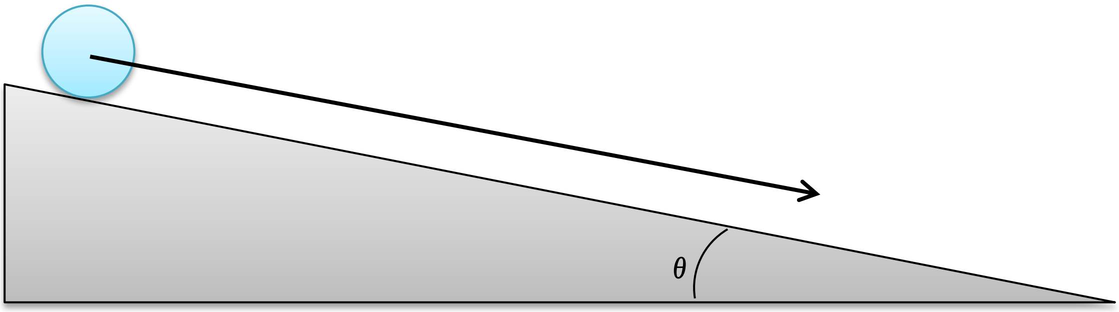
↑跳ね返り係数1

↑跳ね返り係数0



↑跳ね返り係数 0.5 くらい

さて…加速度の話はまだまだ続くよ。次は斜面の話だぜぇ…。例えばボールがあってですよ



図のように斜面の上にボールを置けば当然転がりますわなぁ…とりあえず摩擦がないとした場合について次の問に答えて下さいよぉ…。

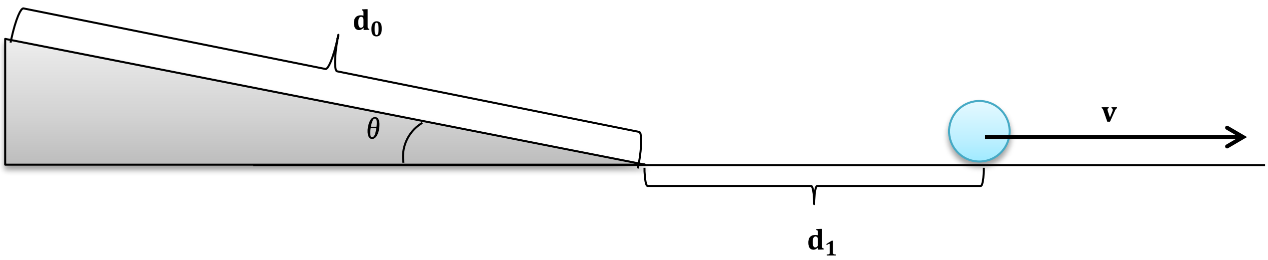
重力加速度を g とし、斜面の角度を θ とした場合

(1)ボールは当然加速しつつ進んでいきます。置いてから t 秒後のボールの速度を求めなさい

―――

(2)置いてから t 秒後に x 方向に移動した距離を求めなさい

―――

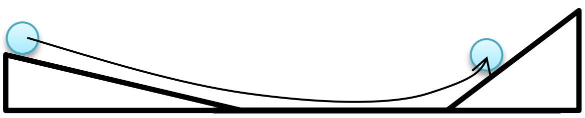


図のようにボールは斜面をd_0進み、そこからは水平に直進した…とする。

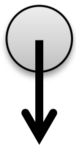
(3)斜面を降りきってからd_1進んだ時のボールの速度 v を求めよ

―――

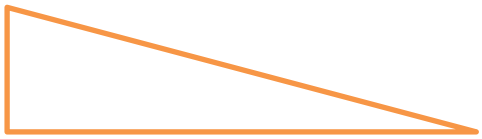
(4)さらに向かい側に角度Ψの斜面があり、ボールは勢いで登っていった、水平面からどれくらいの高さまで登るか考えなさい



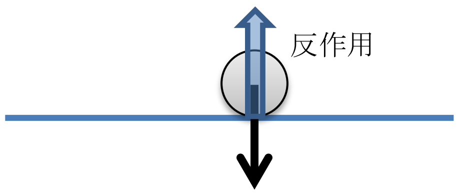
前の問題についてですが、坂道を降りる時の速度についてですが、考え方としては



物理的な力としては「下に降りたい」わけで、重力の方向は真下向きなわけです。
ところが、障害物がありますよね？

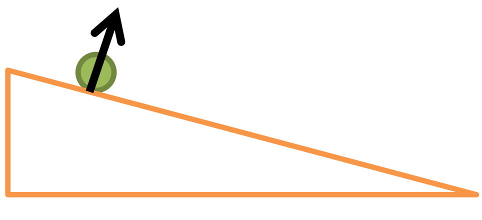


坂道です。常識的にというか、経験的に考えて、動く物体は障害物にあたったら停止しますよね？これを物理学では「作用反作用の法則」という考え方で対応しています。これはなんとなくごぞんじですよね？
例えば地面に対してであれば、常に重力がかかっているのに皆さんが地面にめり込まないのは反作用がかかっているからです。



ゆーたら地面からの反作用で、重力が打ち消されているわけです。地面が水平な場合は全打ち消しです。

では地面が傾いてるなら(坂道なら)どうなるのか？とりあえず、地面に垂直な方向に反作用がかかります。要は「そっち方向に進めない」イコール「反作用がかかってる」と考えるわけ。
そうするとこのように斜面の法線方向に反作用が発生していると考えられます。



摩擦とか抵抗を考えなければ基本的には元の高さまで戻ります。
なんでかというと元々速度０の物体が坂を転がる加速度によりｖになったとしま

す、そしてもう一方の坂を登っていき、最終的には加速度と釣り合ってそこで速度は０になります。
とりあえず最初の坂の加速度をａとし、次の坂での加速度をｂとします。

$v = at_a$なのはわかりますね？t_aはｖに到達するまでの時間つまり坂を降りきった時の時間です。
で速度がつりあう時間をt_bとすると
$v-bt_b=0$ つまり$v = bt_b$です。つまり、$at_a = bt_b$です。

ちなみにａ側の高さは$\frac{1}{2}\mathrm{asin}\theta\ t_a^2$ですね？この高さをひとまず$h_a$としておきましょう。

$h_a = \frac{1}{2}\mathrm{asin}\theta\ t_a^2$ この式を一旦覚えといて下さい。ではｂの高さをh_bとすると

$h_b = \frac{1}{2}\mathrm{bsin}\psi\ t_b^2$です。ここで$at_a = bt_b$ですから$t_b = \frac{a\,t_a}{b}$になりますので$h_b$の式に代入します。そうすると…ちょっとややこしいですが

$$h_b = \frac{1}{2}b\sin\psi\ t_b^2 = \frac{1}{2}b\ sin\psi\frac{a^2t_a^2}{b^2} = \frac{1}{2}\ sin\psi\frac{a^2t_a^2}{b}$$

です。大丈夫ですか？ついてきてますか？ここでａ=gsinθで、b=gsinψですから

$$\frac{1}{2}\ sin\psi\frac{a^2t_a^2}{b} = \frac{1}{2}sin\psi\left(\frac{(gsin\theta)^2t_a^2}{gsin\psi}\right) = \frac{1}{2}\left(\frac{(gsin\theta)^2t_a^2}{g}\right)$$

こうなりますね？ホンマややこしいけどもうちょっとだけ続くんじゃい。

$$\frac{1}{2}\left(\frac{(gsin\theta)^2t_a^2}{g}\right) = \frac{1}{2}gsin\theta^2t_a^2$$

と、ここで思い出して欲しいのがａ=gsinθで逆に言うとgsinθ=ａですから

$$\mathrm{gsin}\,\theta^{\ 2} = \mathrm{asin}\,\theta$$

ですね？つまり

$$h_b = \frac{1}{2}\mathrm{asin}\,\theta\ \ t_a^2 = h_a$$

というわけで同じ高さまで上がってくるわけです。ところがこんな考え方じゃ非常に面倒ですね。というわけで考えだされた法則が**エネルギーとか仕事量**とかいう法則なんですわ。

次は仕事量とエネルギー保存の法則です。

前回に「元の高さまで戻る」ってな話をしましたが、それを見た物理学者が「これってもっと簡単に表現できるんじゃね?」とか考えて法則を考えだしました。

前回の例で言うと坂の角度とかはどーでもいい。とにかく高さが重要なん DA ということで、
特定の高さ h のときのエネルギーを mgh とします。

元々仕事量って意味が
「ある重さの物体をある加速度で動かした距離」を仕事量として定義しています。
mg は重さ×加速度なので、F=ma より、重力により掛かる力を mg とすることができ、特定の高さ h にそれがある時、mgh の仕事量が想定されるわけで、その仕事を行うその潜在的なパワーを「**エネルギー**」と言います。単位は J(ジュール)とか言うやつですが、ゲームプログラマにとっては単位とか基本的には関係無いです。

さて、このエネルギーですが、物理界には「**エネルギー保存の法則**」というとっても大事な法則があり、特に摩擦とか空気抵抗がないならエネルギーの総和(総量)は変化しないという法則があります。

前の坂道の例だと、高さ h にある m の物体は mgh のエネルギーを持っていることになる。

これが保存される…要は総和が一定ということだ。

坂道を転がり切ったとき、高さエネルギーはゼロになっているため、すべてのエネルギーは速さに変換されているはずである。

前の坂道の例を思い出してほしいのだが、坂道方向の加速度は $g\sin\Theta$ である。この加速度でどれくらいの距離を進むのかというと、

$$\frac{h}{\sin\Theta}$$

である。これはわかるよね？

加速度 $g\sin\Theta$ で、$\frac{h}{\sin\Theta}$ の距離を進むのだから

$$\frac{h}{\sin\Theta} = \frac{1}{2}g\sin\Theta\, t^2$$

両辺に $\sin\Theta$ をかけてやると

$$h = \frac{1}{2}g\,(\sin\Theta\, t)^2$$

である。ここで高さエネルギーの定義は mgh だったわけなので、さらに両辺に mg をかけてみる。すると

$$mgh = \frac{1}{2}(g\sin\Theta\, t)^2$$

を得られる。ここで $(g\sin\Theta\ \ t)$ というのは何に当たるのだろうか？そう、下りきった時の速度 v そのものである。

つまるところ最終的には

$$mgh = \frac{1}{2}mv^2$$

という「公式的」なものになる。もしかしたら高校とかで見たことがあるのかもしれない。一応覚えていてもいい公式だと思う。もしかしたらカプコンとか、それ系の試験でも出るのかもしれない。

ともかく、高さが分かっていれば、下りきった時のスピードが一発で計算可能なのである。便利だろう？

では、例えば坂の中間地点…つまり h/2 の時のエネルギー状態ってのはどうなってんだろうね？当然ながら高さエネルギーは mgh/2 である。これの釣り合いが取れるようになるのだからその時の速度を v とすると

$$\mathrm{mgh} = \frac{\mathrm{mgh}}{2} + \frac{1}{2}\mathrm{mv}^2$$

となる。まあ、現在の位置さえわかってれば、その時の速度がわかっちゃうわけね。じゃあもっと一般化して、0<s<h という条件の s を考えて、高さ s の時の速度との関係も当然

$$\mathrm{mgh} = \mathrm{mgs} + \frac{1}{2}\mathrm{mv}^2$$

になるわけだから、

$$\mathrm{mg(h-s)} = \frac{1}{2}\mathrm{mv}^2$$

$$2\mathrm{g(h-s)} = \mathrm{v}^2$$

$$\mathrm{v} = \sqrt{2\mathrm{g(h-s)}}$$

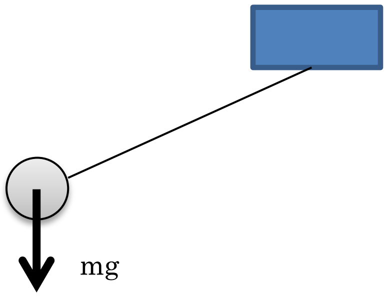
となるわけだ。まあ、便利っちゃあ便利。

ともかくここで重要なのは高さエネルギーと、速度エネルギーを足したものは常に一定ってこと。ここは覚えておこう。

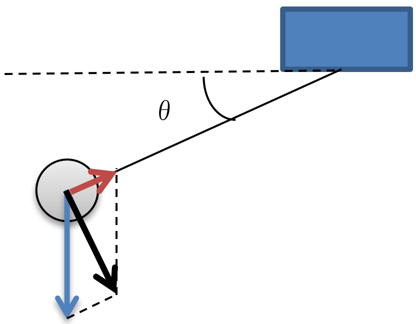
ちなみに振り子運動に関しても、エネルギー保存の法則は適用可能なので、各自考えてはおこう。ただし周期を真面目に計算しようと思わないほうがいい…相当面倒なことになるだろう。不慣れな人が挑戦すると命をなくすことになる。

実際にプログラムで実装するときは sin とか cos を使いつつ近似的に「それっぽい動き」を実装するのが無難であろう。と思います。

ともかく、プログラミング的に順を追って考えていきましょう。
そもそも振り子がなんであいう動きをしているのかを細かく見ていきます
初期位置が下図のようになっているとします。



当然重力が mg かかっているんですが、紐に繋がれているので「張力」ってのが発生します。「反作用」みたいなもんですけど、そのまま g に従って落ちると紐の長さが保てなくなるので、紐の付け根方向に物体を戻そうとする力…それが張力です。



結局、図のようになり、結果として「紐の長さを中心とした円弧運動」を行います。
そして加速度は g かかるべきところが gcosθ となります。

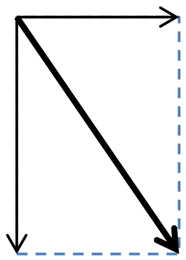
物体が支点から鉛直方向(つまりひもが縦に真っ直ぐの状態)にあるとき、加速度が 0 になります。なぜなら cos90°=0 だから。

そして、最下点を通過すると今度は上に上がっていきますが、それは一度貯めこんでおいた速度が重力で少しずつ削られてまたゼロになっていくわけです。

とりあえずそこまでシミュレーションしてみましょう。重力とひもの長さは適当に自分で設定して下さい。
ちなみに DxLib で線は DrawLine で引けますし、円は DrawCircle で描画できます。

もっというと、cosθは内積を使えば賢く計算できますし、sinθは外積を使用します。まあ、cosθがわかれば、自動的に sinθがでるようにしてもいいですけどね。
もしくは cosθから atan2 関数で角度出しておいて、そこから sinθを計算してもいいでしょう。

なぜ僕が cosθだけでなく sinθのことも話しているのか？
それはこういうことだ。cosθで接線方向の速度が分かる話はしましたね？

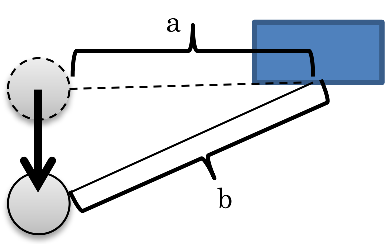


実際の運用上、それを x 方向と y 方向に分かる必要があるわけですから、そこで sin も cos も必要になるわけです。

以前の『勾配』との違いは、位置が変わるたびにこの角度が変わっていく…そう考えるとなんとか作れそうでしょ？

ちなみにここまでやるためには、ひもの端点の位置は覚えておかないといけないので、注意してね。

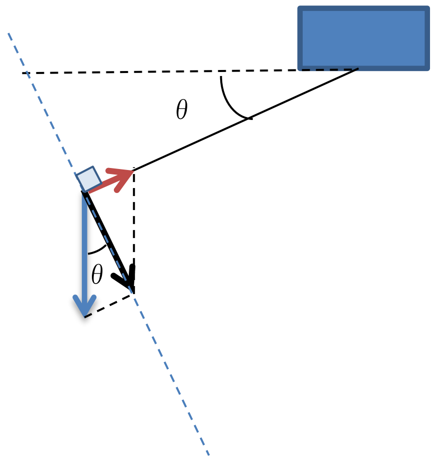
結局のところ、上のようなやり方で現在の角度で gcosθの加速度を加えていくことになるわけですが、ここで注意点…
例えば図のように水平が初期状態の場合 g がそのままかかるわけで、そのまま加速してしまうと



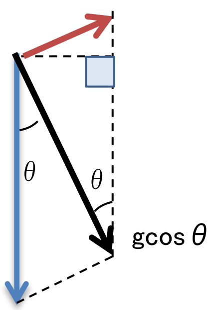
図のような状態になり見ての通り a<b という状態になります。とりあえず「ゴム」ではないので a=b を保たねばならないわけです。
なので、加速度で移動した上で長さを補正してください。「長さを補正」とは b の長さを a にする処理で、方向を保ったまま長さだけ a にする。つまり…あとはわかるな？

上では通常の加速度がひもによって制限される観点から話をしました。

最終的にオモリは「オモリと支点を結ぶベクトルに直交する方向に加速する」わけですから、
こうなります。



そうなると、前にも書きましたが、加速度 gcosθ が発生するわけです。a=gcosθ ですね。プログラムでいうところの速度と加速度の式は
v=v+a;
ですから、これによって速度がどんどん上がっていくのが分かります。
あとはこの加速した v を x 方向ベクトルと y 方向ベクトルとに分けることにより、実際の X 座標 Y 座標を更新します。ですから、
x+=cosθ*v;
y+=sinθ*v;
ですね。なぜならば



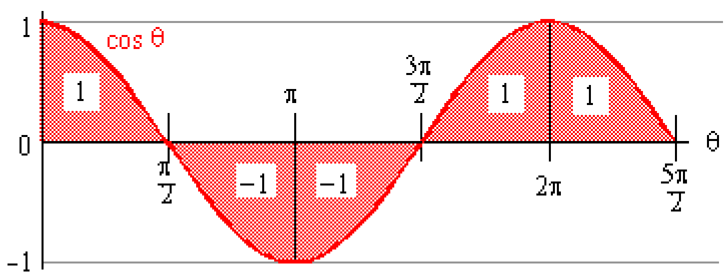
となるからです。流石にもう図をみて分かるレベルになっていると信じますが…
分からない人は言って下さい。

と、ここまでやれば振り子運動はできるようになるかと思います。
なお、ただ振り子運動をするだけであれば、実は今回の話はもうちょっと簡略化できます。微分積分を知っていると「加速度は速度の積み重ね」って分かるわけですから

$$速度 = \int g\cos\theta \, d\theta$$

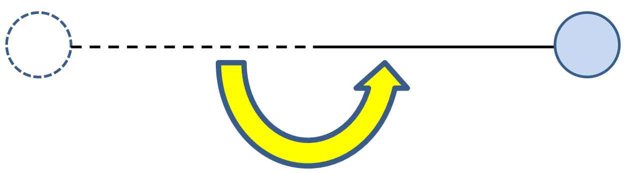
なわけです。ここで覚えておいて欲しい記号は∫って記号。これは「インテグラル」と言って、「積分」を表すものです。
積分ってのは読んで字のごとく「分けたものを積み重ねる」って意味です。もっと言うと「微細に分けたものを積み重ねる」って意味です。

ところで積分ってのは簡単に言うと「面積」です。通常の面積であれば縦×横ですぐ計算できるのですが、今回の $g\cos\theta$ のように値がガンガン変わっていくものの面積を計算する場合、そう簡単に行きません。だから積分を使うのですが、$g\cos\theta$ の面積のイメージは



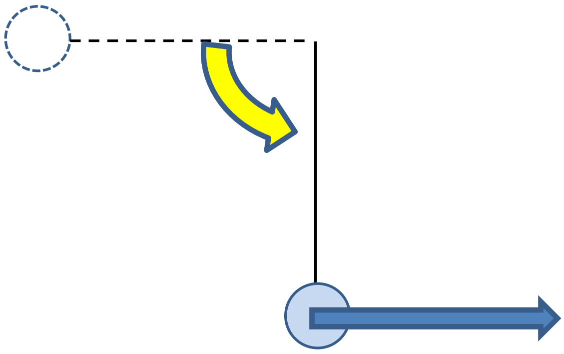
こんな感じで、だんだんだんんだん面積を積み重ねていって、一旦値がマイナスに入っちゃいますので、ここは面積といえども面積値が減る計算になります。つまり速度としてはπになったところでいったんゼロになります。

振り子の状態で言うと



この状態。速度がゼロになっているでしょう？逆にπ/2 の時は加速度が 0 ですが、

速度が最大になります。



こんな感じですね。で、ここで話を戻しますがぶっちゃけ答えだけ言うと

$$速度 = \int g\cos\theta\, d\theta = g\sin\theta$$

です。つまり加速度すっとばして

v=gsinθ

とすることができます。つまり

x+=gsinθ*cosθ

y+=gsinθ*sinθ

でも別にいいわけです。

ただし、このやり方の場合は「力」を無視しているので「やまめすいんぐ」は作れませんけどね。

「やまめすいんぐ」の作り方は、とりあえずきちんと「加速度」を意識した作り方のアルゴリズムをで作った上で、3つのフェーズを作ります。

① 初期位置から真下に加速するフェーズ
② 左右に振っていって、振り子の振幅を上げていくフェーズ
③ ひもを切ってすっ飛んでいくフェーズ

まあ、優秀なる皆様のことでございますから①は別に解説の必要は無いでしょう。

②に関してですが、これは「近似版」で実装すると逆に難しくなるので、最初の「加速度版」で実装しましょう。なんで近似版で実装しようとすると難しいのかというと、左右に振るってことは、力がかかるわけです。

$$F=ma$$

この式により「力」が表されますが、つまるところ加速度がかかるということです。