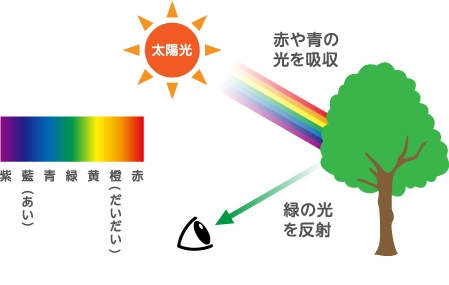
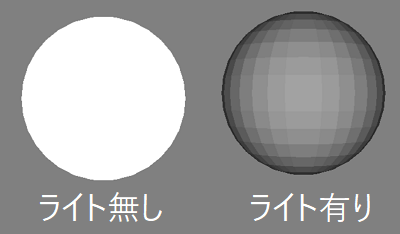
01 – ライティング基礎

**物体を認識する仕組み**

　人間の目は、飛び込んだ光の量に応じて色を認識します。真っ暗闇では光が飛んでいないため、黒色と認識します。日中では太陽の光が様々なものに反射して、人間の目に飛び込んできます。光は目に見えないですが、光の加法混色から赤色・緑色・青色の光と分けることができます。赤く見える物体というのは、赤色の光だけを反射し、緑色・青色の光は物体に吸収されます(反射しなかった光が貫通するのであれば、影はできないはず)。また、日中に黒色に見える物体というのは、光を反射せずに吸収する物体といえます。

※補足として、宇宙空間が黒色なのは反射する物体がないためです。吸収されていると地球に光が届かないことになります。また空が青いのは、光が空気中の粒子にぶつかり、青色の光を反射しているからです。

　コンピューターグラフィックスでは、色を付けるだけでは3Dを表現できません。物体の光の反射で白く光ったり、影が落ちることで暗くなったりと、陰影をつけることでより立体感が増します。ゲームではこの計算をピクセルシェーダーで行います。ゲームエンジン任せになっていた光の計算を実際に行ってみましょう。

**Lambert(ランバート)シェーディング**

ダイアグラム

自動的に生成された説明直接オブジェクトに当たる光のことを「直接光」といいます。Lambertでは、物体の表面に当たる直接光の反射を計算します。この反射した後の光を「拡散光」(Diffuse)といいます。

この計算では「オブジェクトの法則」と「光源の進行ベクトル」との

ダイアグラム

自動的に生成された説明「内積」から光が当たる強さを計算します。光の反射としては物理的に正しくないですが、高速でそれっぽく見える計算です。内積の計算結果は、同じ方向を向くベクトル同士なら「1」、直交するベクトル同士なら「0」、反対方向を向くベクトル同士だと「-1」となります。

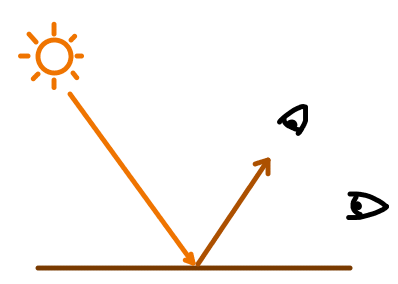
Lambertでは、この内積の性質を利用して反射光の強さを計算します。ただし、この計算は単位ベクトル同士で計算したものなので、それぞれのベクトルを正規化したうえで計算する必要があります。また、そのまま計算を行うと、光の当たる面が「影」となってしまうため、計算の際には‘あえて’ライトの方向ベクトルを反転させます。

**Ambient(アンビエント)**

ダイアグラム

自動的に生成された説明　先ほどのLambertで表現できる光は、オブジェクトに直接あたる光だけです。しかし現実世界では、様々な物体から反射した間接的な光(間接光)を受けて、直接光が当たっていない部分も明るくなります。この間接光が反射した後の光を「環境光」(Ambient)といいます。

　正確に計算しようとすると、複雑な反射を計算しなければならないため、ゲームではどのオブジェクトも一定量の間接光を受けているとみなして、それっぽく見えるように計算します。これらを踏まえて、ランバートシェーディングの計算は、光の加法混色から「Diffuse」と「Ambient」を足し合わせた結果となります。



**Phong(フォン)シェーディング**

　現実世界には、金属や鏡のように強く光を反射する物体が存在します。これらの物体は常に光が強く飛び込んでくるわけではなく、ある一定の角度の時だけ反射した光が強くなります。物体表面に当たった光は一定の角度で反射するため、目の位置に応じて光の強さが異なります。この物体表面で強く反射する光のことを「鏡面反射光」(Specular)といいます。

グラフ

自動的に生成された説明　この鏡面反射を計算するためには、ライトの方向ベクトルだけでなく、「反射ベクトル」を求めなければいけません。このベクトルはぶつかって跳ね返った時の方向を示します。

ダイアグラム, 概略図

自動的に生成された説明R = 反射ベクトル / L = ライトの方向ベクトル / N = 物体の法線

この計算で出てきた内積には、先ほどとは異なる性質を計算に利用しています。一方のベクトルが単位ベクトルの時、他方のベクトルが単位ベクトルに落とす影の長さと等しいというものです。あるいは3Dオブジェクトを平たく潰す(Zスケール0にする)と2Dになるように、2Dの情報を1D（線）に落とし込む計算と言ってもいいでしょう。どちらにせよ、イメージしづらいかもしれませんが、シェーダーに限らず3Dの計算においては非常に重要な性質です。

話を戻して、先ほどの式で反射ベクトルが求められるのか、L(-1, -1)、N(0, 1)とおいて計算してみます。

反射ベクトルの式に値を当てはめる

-Lの部分に-の値をかける

内積の式を立てる

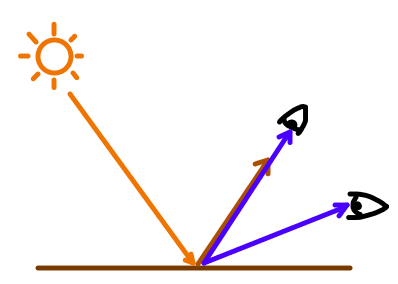
内積を計算する

内積の結果と定数２をかける

定数と法線をかける

先の結果とライトの方向ベクトルを足す

ライトベクトルが反射したベクトルだと求めることが出来ました。ただ、HLSLにはこの反射ベクトルを計算してくれるreflectという関数があるので、こちらを利用してもよいでしょう。ただし、この関数の引数に渡すライトの方向は、本来の方向を渡さないといけないので、あえて反転したライトの方向ベクトルを渡してと、おかしな結果になってしまうので注意してください。



反射ベクトル計算後、光が入射した座標から視点に向かって伸びるベクトルを求めます。この２つのベクトルの向きが近ければ近いほど、反射した光が大量に飛び込んでくることになります。ここでもまた、内積を利用して同じ向きなら１、離れるとどんどん小さくなるような数値を求め、この結果を鏡面反射の強さとします。

最後に、この計算結果のままではかなりの範囲が明るいことになるため、範囲を絞る必要があります。全体的に数値を落とすだけでは明るく見えている部分も暗くなってしまうため、明るい部分は明るいまま、反射ベクトルから外れている部分がより暗くなるように計算します。この計算では累乗を利用します。

　また、鏡面反射は強く反射する部分の明るさのみを計算するだけなので、先ほどのLambertの計算と合わせて、影の計算を行う必要があります。