自作アルゴリズムの開発とその考察

動機

ルービックキューブには43,252,003,274,489,856,000のパターンがありその膨大なパターンの中でも20手以内に必ず解けます。最大なパターンの中でから最適解を見つけ出すことができるアルゴリズムを作りたいと思い今回のアルゴリズムを開発した。

バージョン1の内容

一番初めの状態がA,目標の状態がB,
$\overrightarrow{AB}$と選択肢$\overrightarrow{b_n}$のなす角が一番小さくなる選択肢を選ぶというアルゴリズム

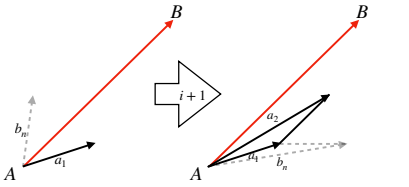
cは繰り返しの実行回数,
dはb_nの選択肢の数

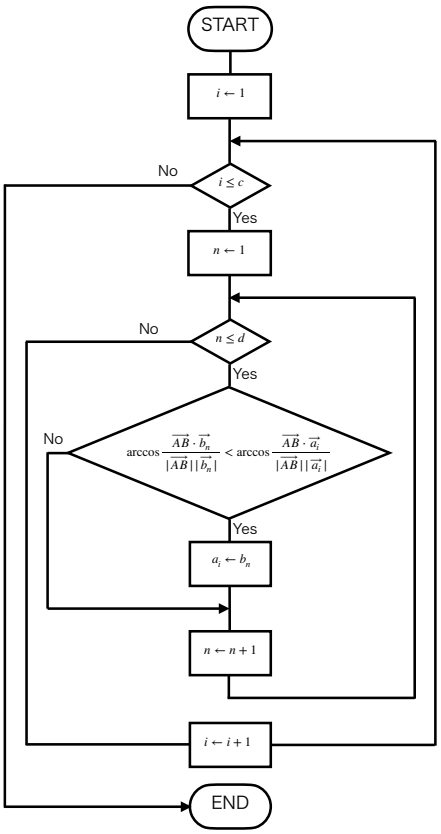
$$\arccos\frac{\overrightarrow{AB}\cdot\overrightarrow{b_n}}{|\overrightarrow{AB}||\overrightarrow{b_n}|},$$

$$\arccos\frac{\overrightarrow{AB}\cdot\overrightarrow{a_n}}{|\overrightarrow{AB}||\overrightarrow{a_n}|}$$

は2ベクトルのなす角を求める式

選択肢$\overrightarrow{b_n}$と$\overrightarrow{AB}$のなす角が一番小さくなる$\overrightarrow{b_n}$を選ぶ。
a_iには選ばれたb_nが代入される。

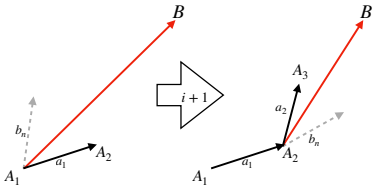


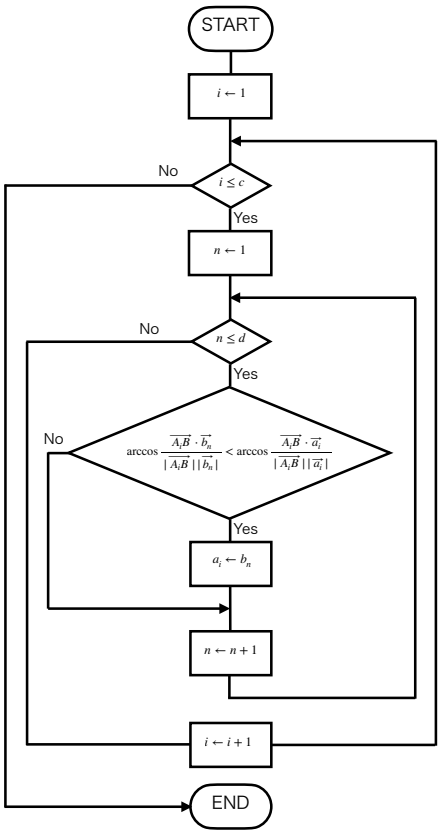


バージョン2の内容

$i \leftarrow i+1$の時にAの位置を更新するためにA_iというベクトルを格納した配列を使うことにした。

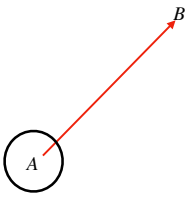
そのためc, d, i, nなどは前と変わらないが$\overrightarrow{AB}$は$\overrightarrow{A_iB}$に置き換わり
一番初めの状態がA_1,
目標の状態がB,
選択肢$\overrightarrow{b_n}$と$\overrightarrow{A_iB}$のなす角が一番小さくなる$\overrightarrow{b_n}$を選ぶというふうになる。





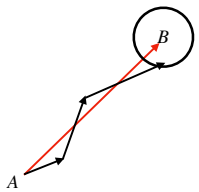
バージョン1の結果

実行してみると
$\overrightarrow{a_i} = -\overrightarrow{a_{i+1}}$となり$A$付近で行ったり来たりを繰り返した。
$\overrightarrow{a_i} = -\overrightarrow{a_{i+1}}$をできないようにするとベクトルの角だけなるべく小さくしてBに近づこうとせずA付近にベクトルが密集した。



バージョン2の結果

実行してみると
バージョン1のA付近にベクトルが密集する代わりに
今度はBに近づいてから
B付近にベクトルが密集した。



考察、今後の展望

今回は一番ベクトルのなす角が小さくなる選択肢のみを選ぶようにしていたが、
一番ベクトルのなす角が小さくなる選択を選ぶだけだと
ベクトルが密集してしまったり詰んでしまうようになってしまった。
それでは目的の状態にたどり着くことが難しいので、
一番小さくなる選択肢以外も実行できるように探索木に応用してみたい。