世界大会対策講座

立命館大学情報理工学部プロジェクト団体RoboCup SimulationLeague部門 Ri-one

発表者：岡崎 翔
協力：中川 貴裕
小栗 真弥
高下 雅洋
三宅 皐

Ri-oneの紹介

三部門で構成される立命館大学情報理工学部プロジェクト団体のうちの一部門

立命館大学情報理工学部プロジェクト団体

RoboCupSimulationLeague部門Ri-one

設立：2005年4月

参加：レスキューSim，2Dサッカーサッカー

部員：8期生7人，9期生6人

http://rione.org/blog/

実績：	2DSoccer	RoboCup2006 in Bremen	3位
	Rescue	RoboCup2009 in GRAZ	7位
	Rescue	RoboCup2010 in SINGAPORE	4位
	Rescue	RoboCup2011 in ISTANBUL	7位
	2DSoccer	RoboCup2011 in ISTANBUL	13位
	Rescue	**RoboCup2012 in MEXICO**	**優勝**

発表内容

開発に使用しているツールなど

- eclipse
- subversion
- テスト

アルゴリズム

- エリア分割(AreaPartition)
- 瓦礫除去優先順位(CriticalArea)
- 通行可能判断(POV)

開発に使用しているツールなど

eclipse

subversion

テスト

開発に使用しているツールなど

eclipse

eclipseとは

- 統合開発環境である
- 多数のプラグイン機能がある
- プラグインにより，Java以外の言語にも対応できる

eclipseを使う理由

利点

- 多数のプラグインがあり，拡張できる
- 補完，エラー部分の表示など，コードの編集支援機能を持っている
- Linux ,Windows ，Macのどれでも動作する

eclipse プラグイン

Findbugs

- コンパイラが通っても残っているバグを見つけてくれる

subversive

- バージョン管理システムsubversionを使える(次スライドで説明)

開発に使用しているツールなど

subversionとは

・無償で使えるバージョン管理ソフト

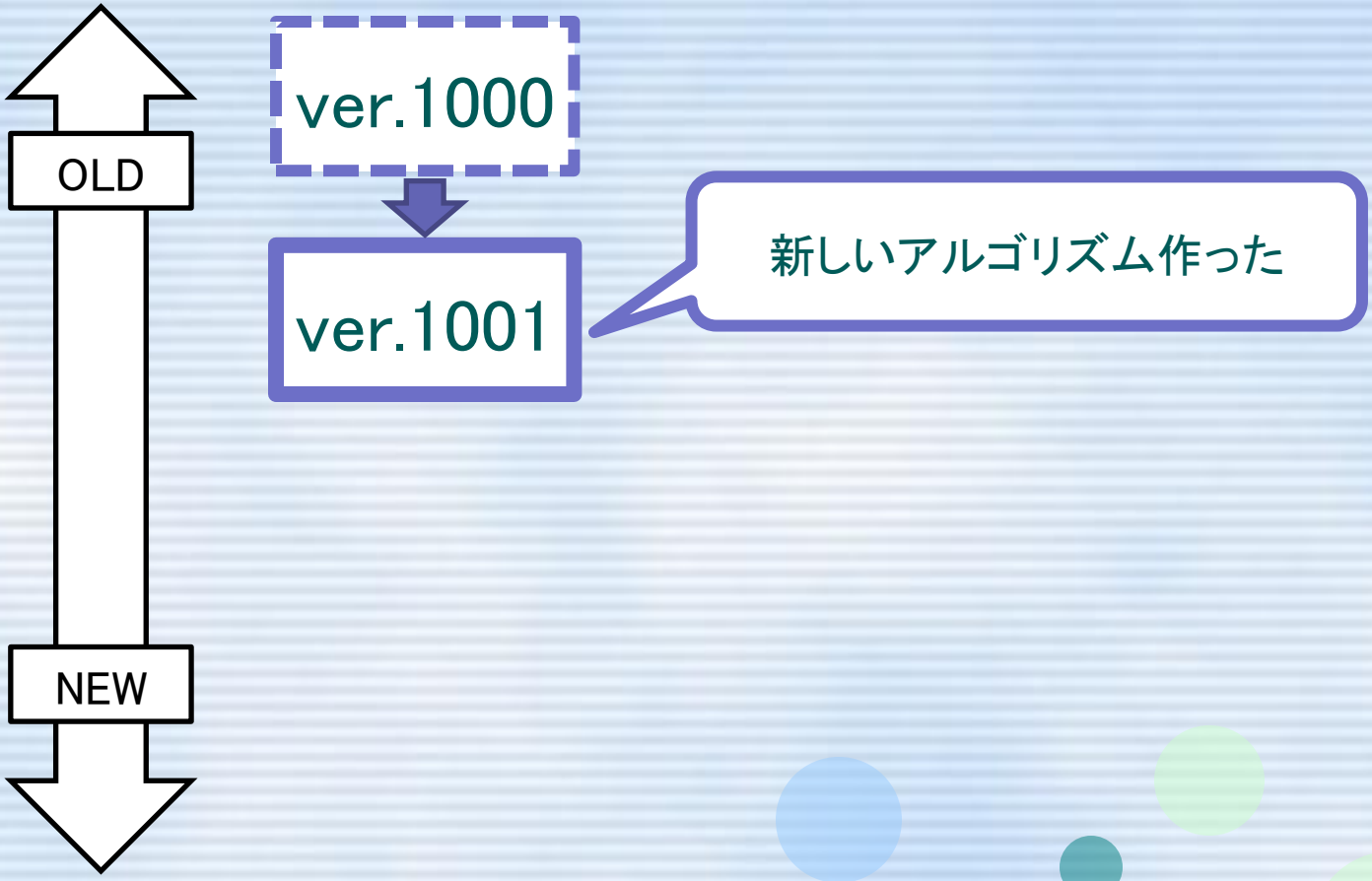
バージョン管理ソフト

・ファイルの変換履歴を管理できるので，以前のバージョンに簡単に戻ることができる

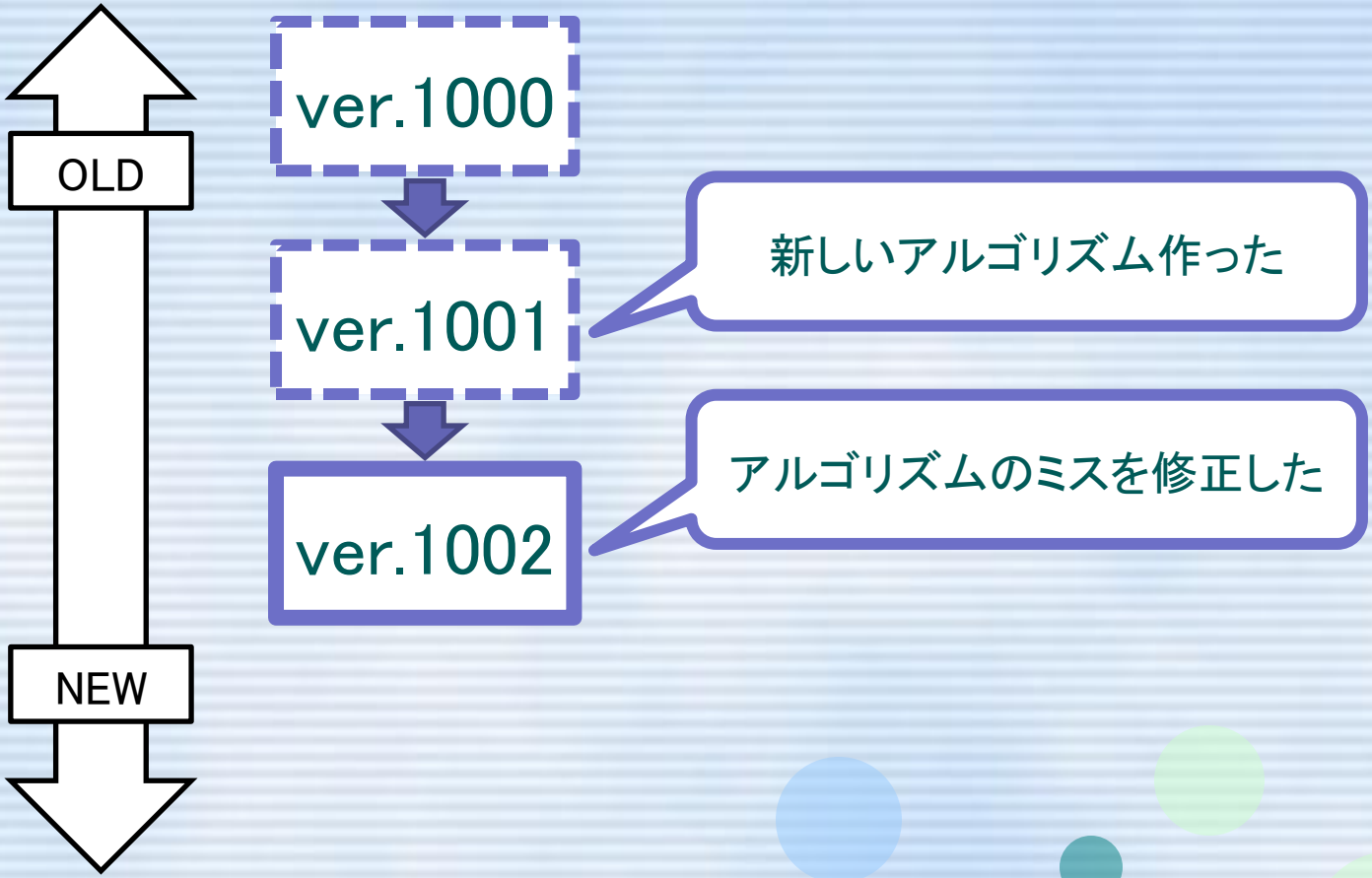
subversion（バージョン管理ソフト）

ver.1000

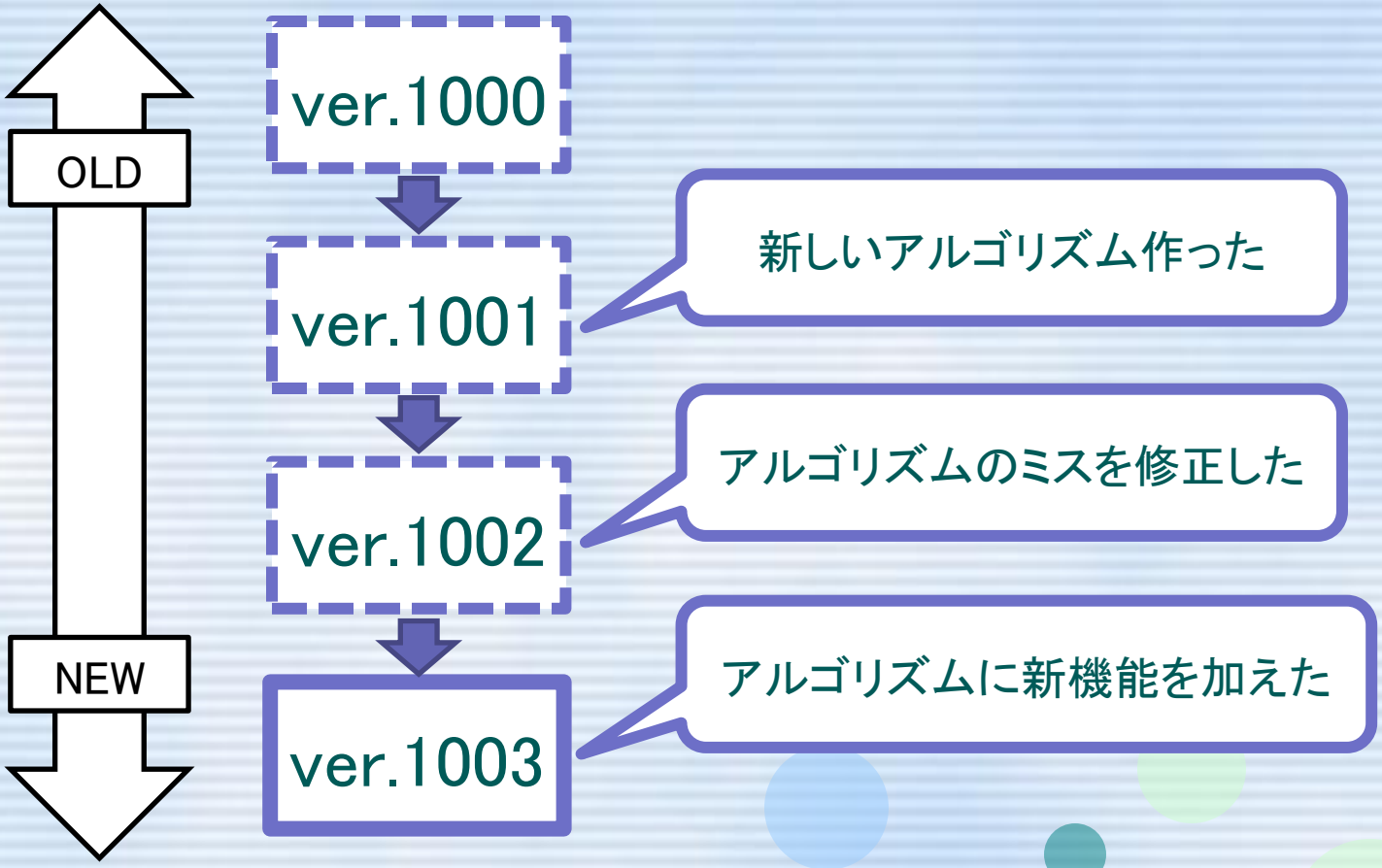
subversion（バージョン管理ソフト）



subversion（バージョン管理ソフト）



subversion（バージョン管理ソフト）



subversion（バージョン管理ソフト）

ver.1000

ver.1001

新しいアルゴリズム作った

ver.1002

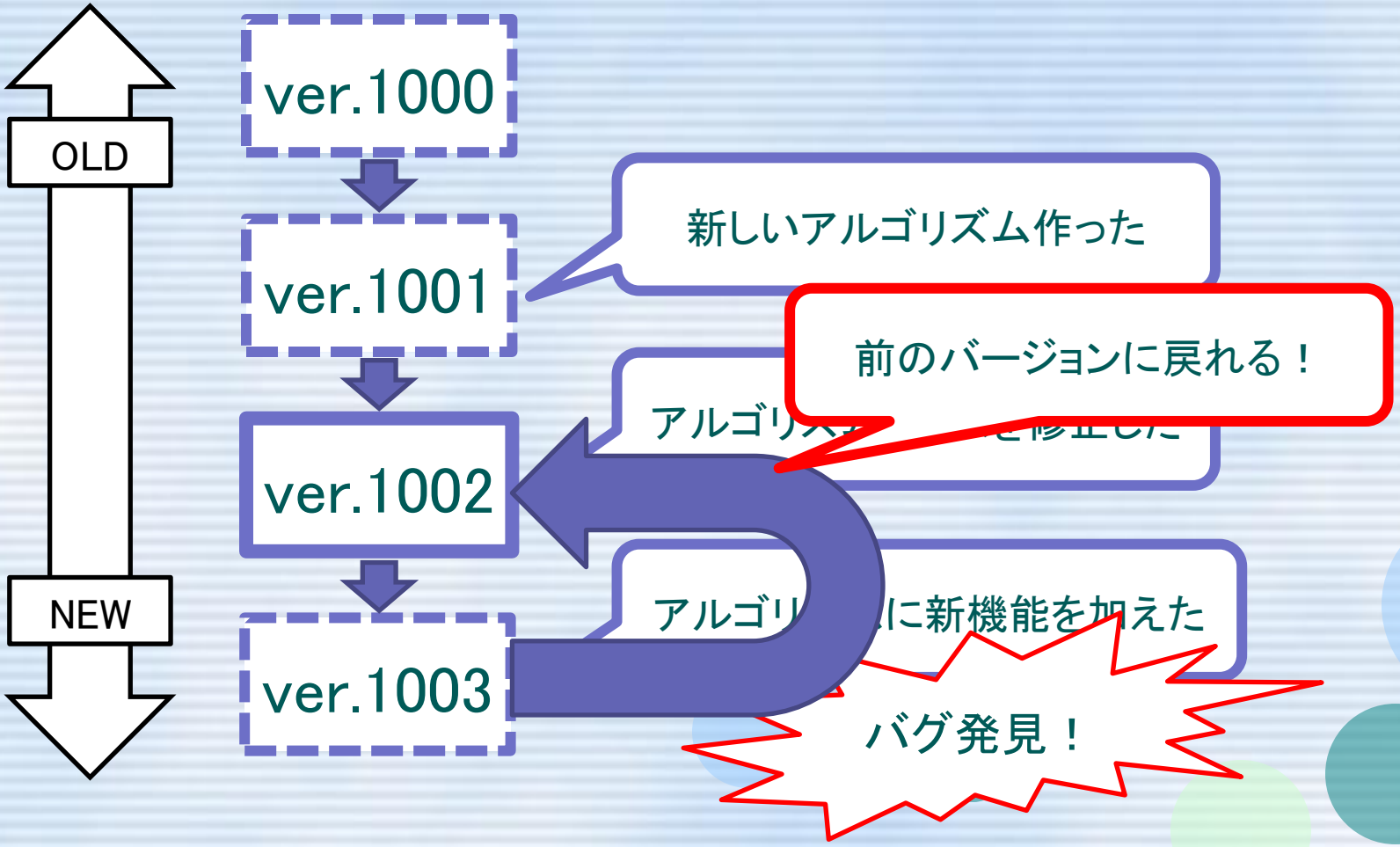
アルゴリズムのミスを修正した

ver.1003

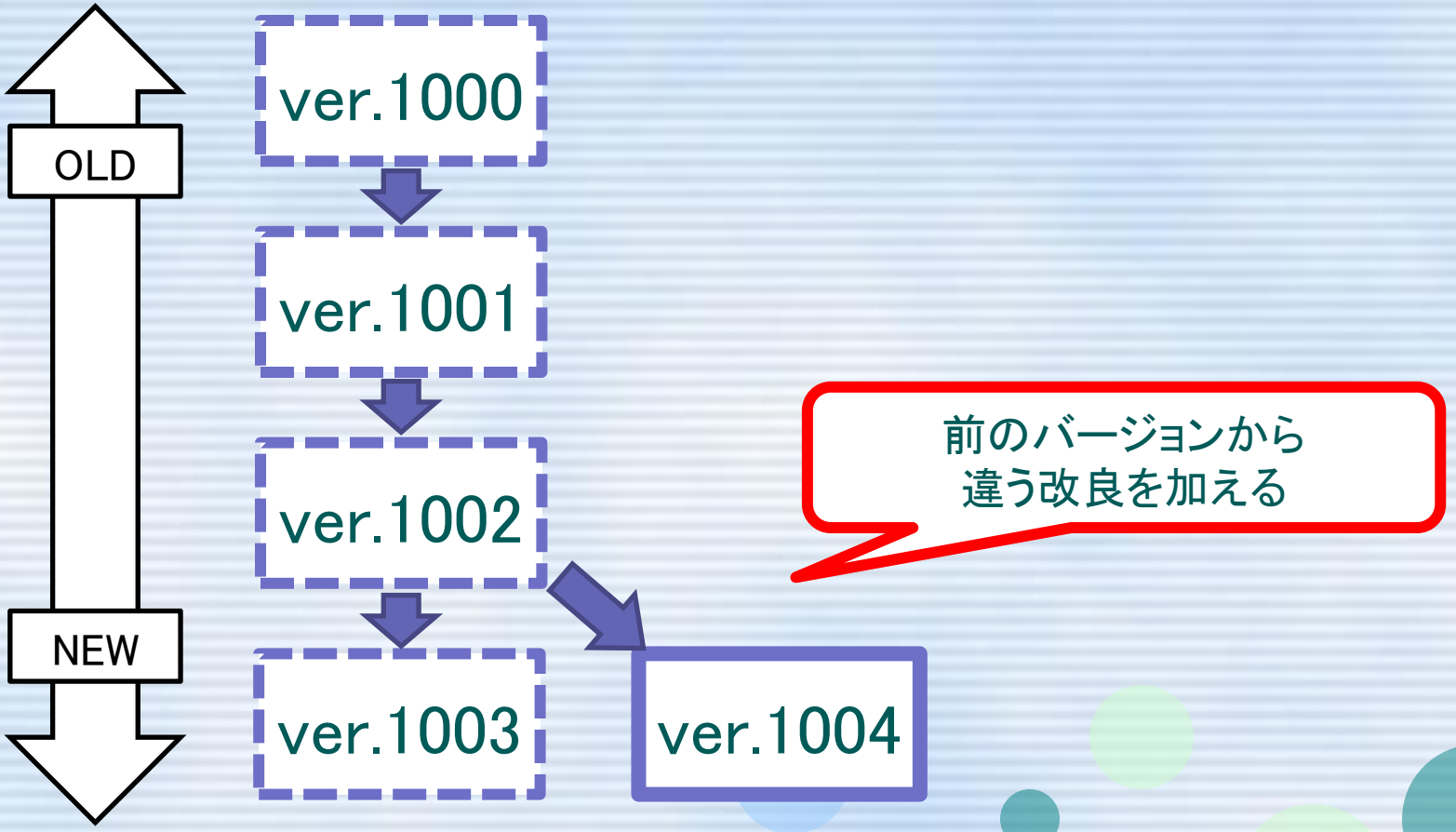
アルゴリズムに新機能を加えた

バグ発見！

subversion（バージョン管理ソフト）



subversion（バージョン管理ソフト）



subversion

バージョン管理のメリット

- 開発したアルゴリズムが上手くいかなかったとき，戻すことが容易
- eclipseと組み合わせてバージョンごとの比較ができる

もし使わなかったら

- 作ったソースをメールやUSBや印刷物で渡す？
- 一回消したら戻れない

開発に使用しているソフトなど

テスト

テストとは

- Ri-oneの勝因の一つ？
- 各バージョンで何回も実行し結果を記録する

テストのメリット

- 多くの結果から比較することで，一番良いバージョンを選び出す

テストで注意したこと

様々なマップを複数回ずつ実行し，

その結果をすべてメモする

もし以前より点数が下がっていたら，

比較をして修正すべき点を見つける

点数だけに注目するのではなく，

各エージェントの動きにも注目する

テストで注意したこと

様々なマップを複数回ずつ実行し，

その結果をすべてメモする

Ri-oneは何回もテストを行い一番良いバージョンを選び，世界大会に使用した

点数だけに注目するのではなく，

各エージェントの動きにも注目する

アルゴリズム

- AreaPartition
- CriticalArea
- POV

アルゴリズム

- AreaPartition
- CriticalArea
- POV

AreaPartition

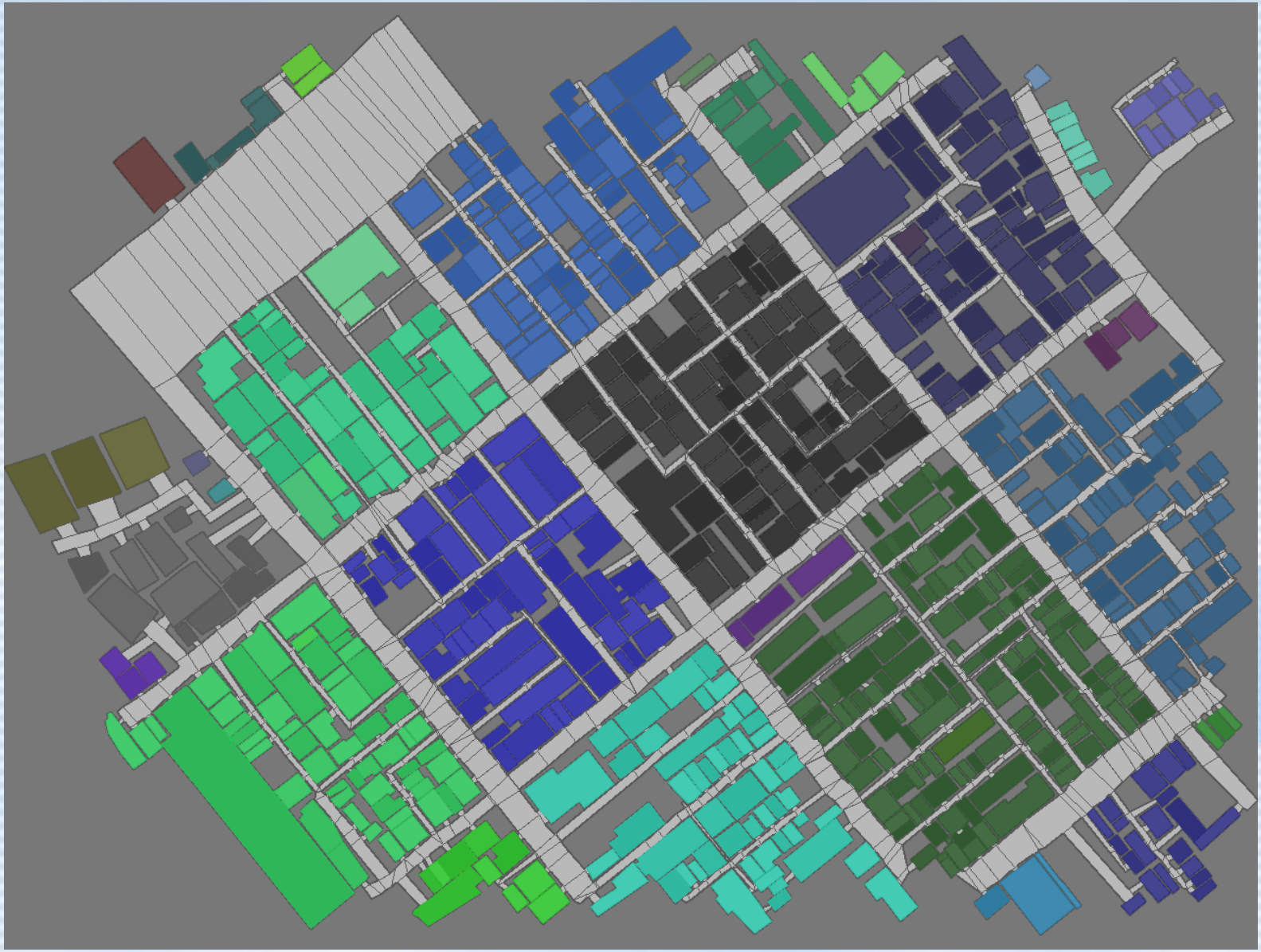
内容

- 建物の集合を区画の一つとする
- 消火や救助に利用
- 少ない情報量で通信が可能

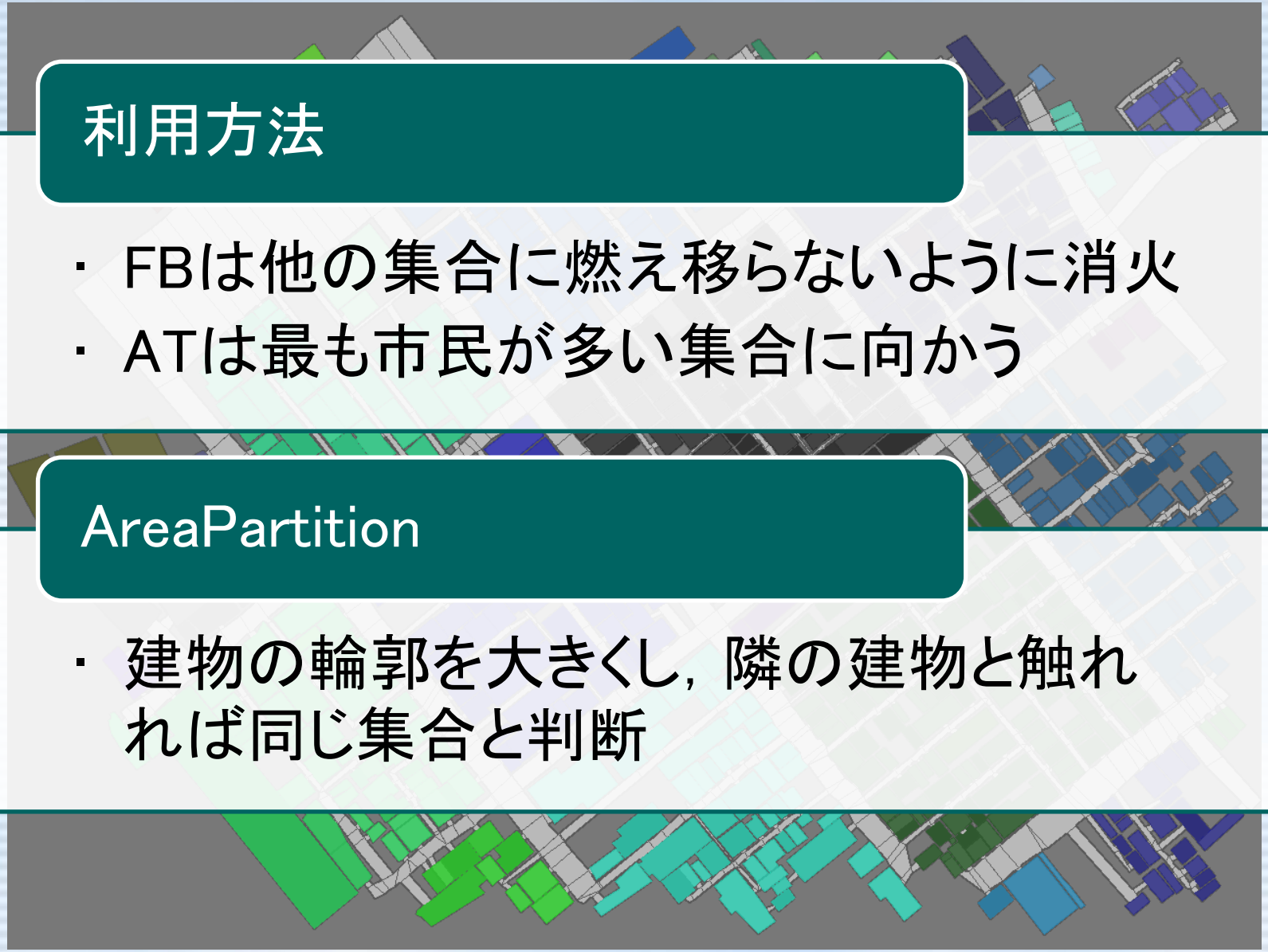
目的

- 集合単位でエージェントに思考させる
- 通信量の節約

AreaPartition



AreaPartition



利用方法

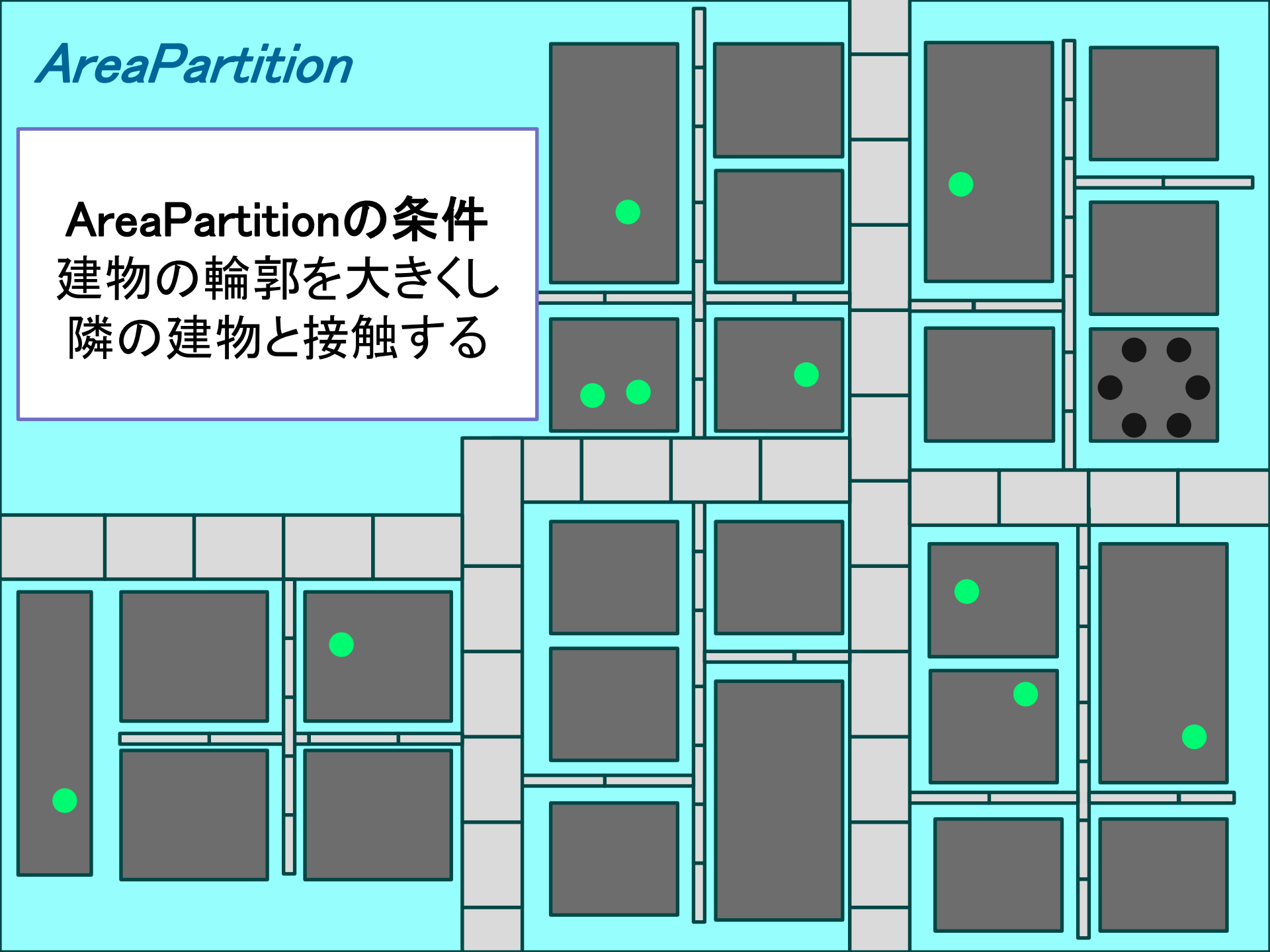
- FBは他の集合に燃え移らないように消火
- ATは最も市民が多い集合に向かう

AreaPartition

- 建物の輪郭を大きくし，隣の建物と触れれば同じ集合と判断

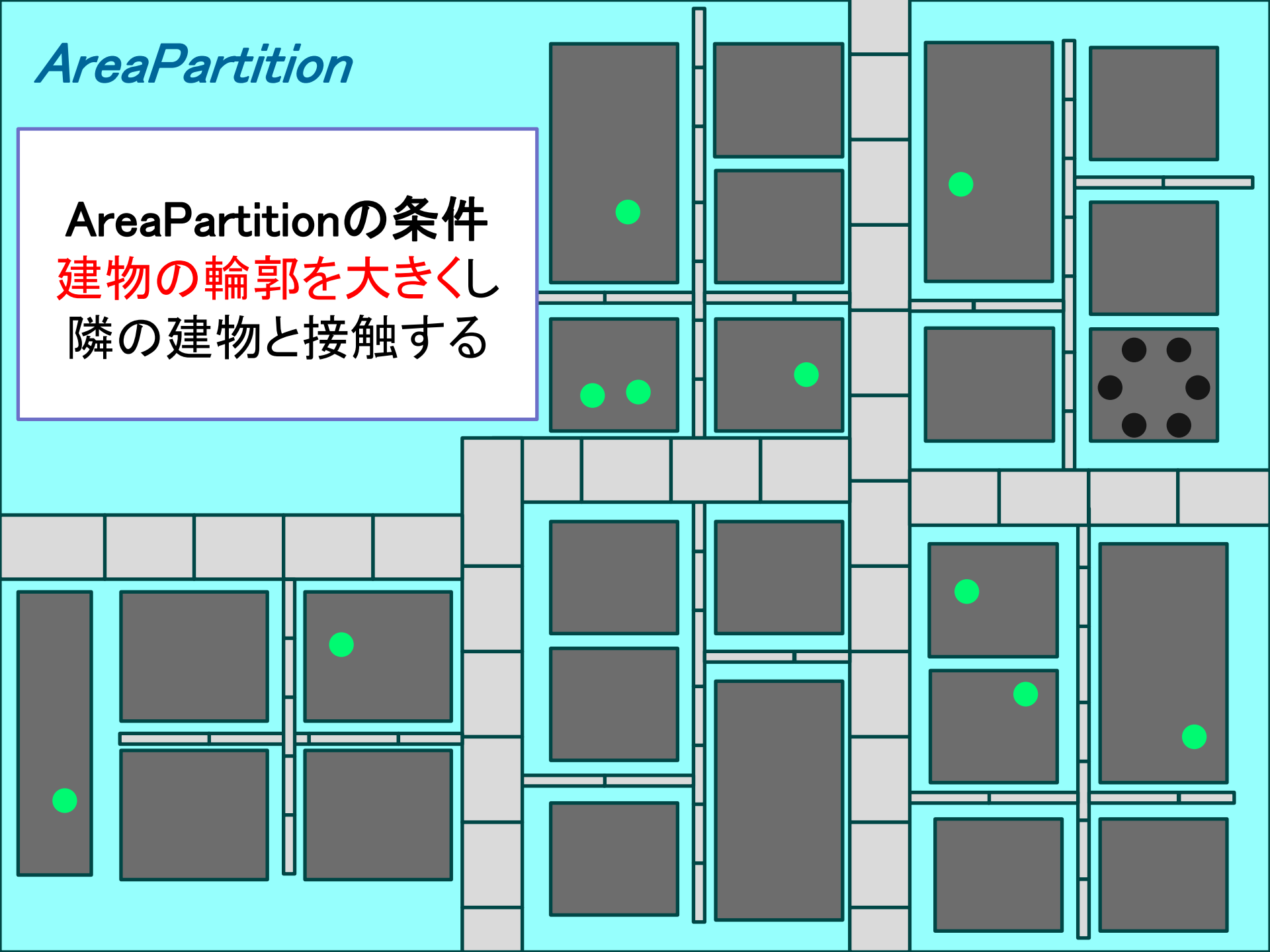
AreaPartition

AreaPartitionの条件
建物の輪郭を大きくし
隣の建物と接触する



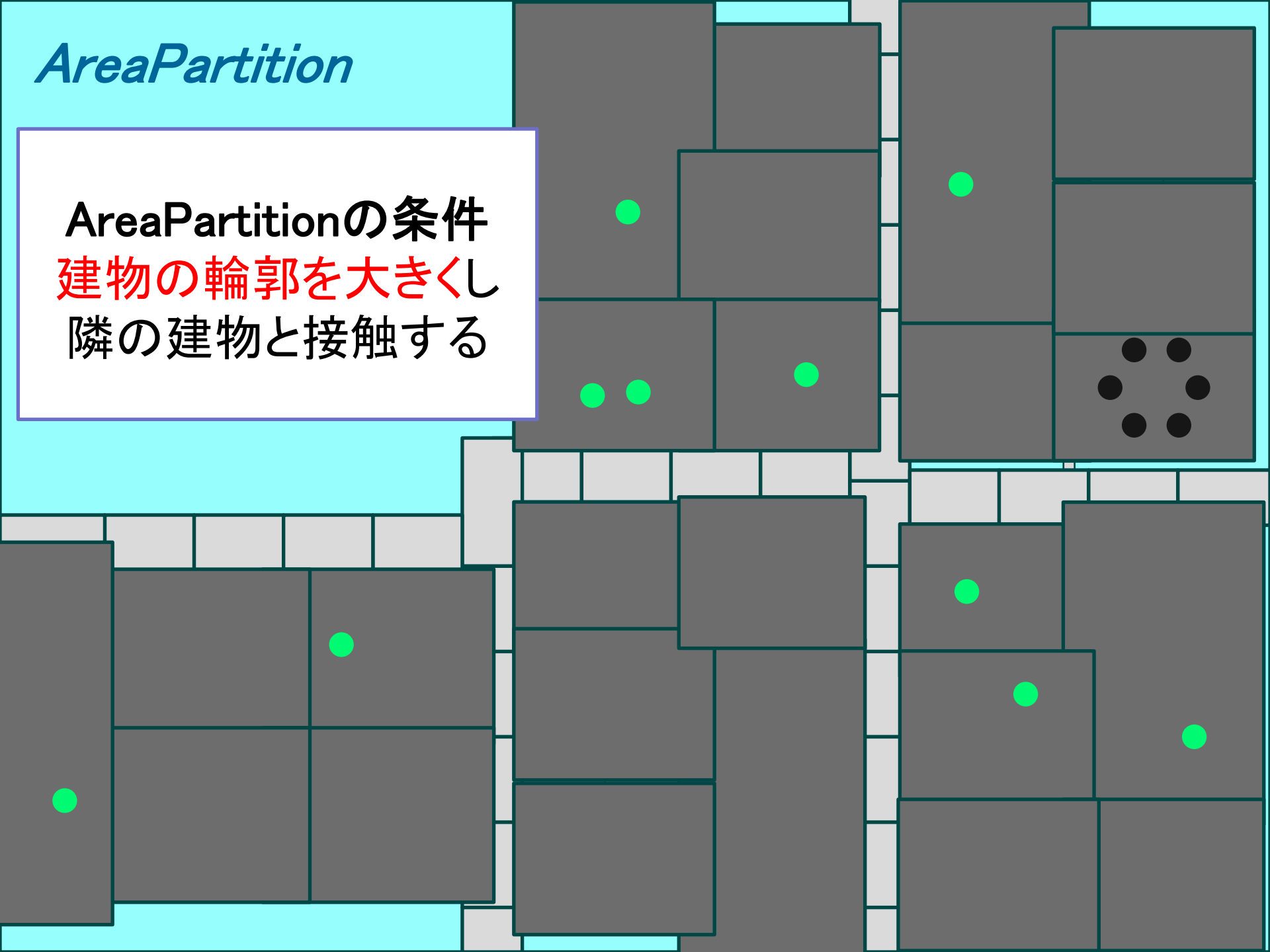
AreaPartition

AreaPartitionの条件
建物の輪郭を大きくし
隣の建物と接触する



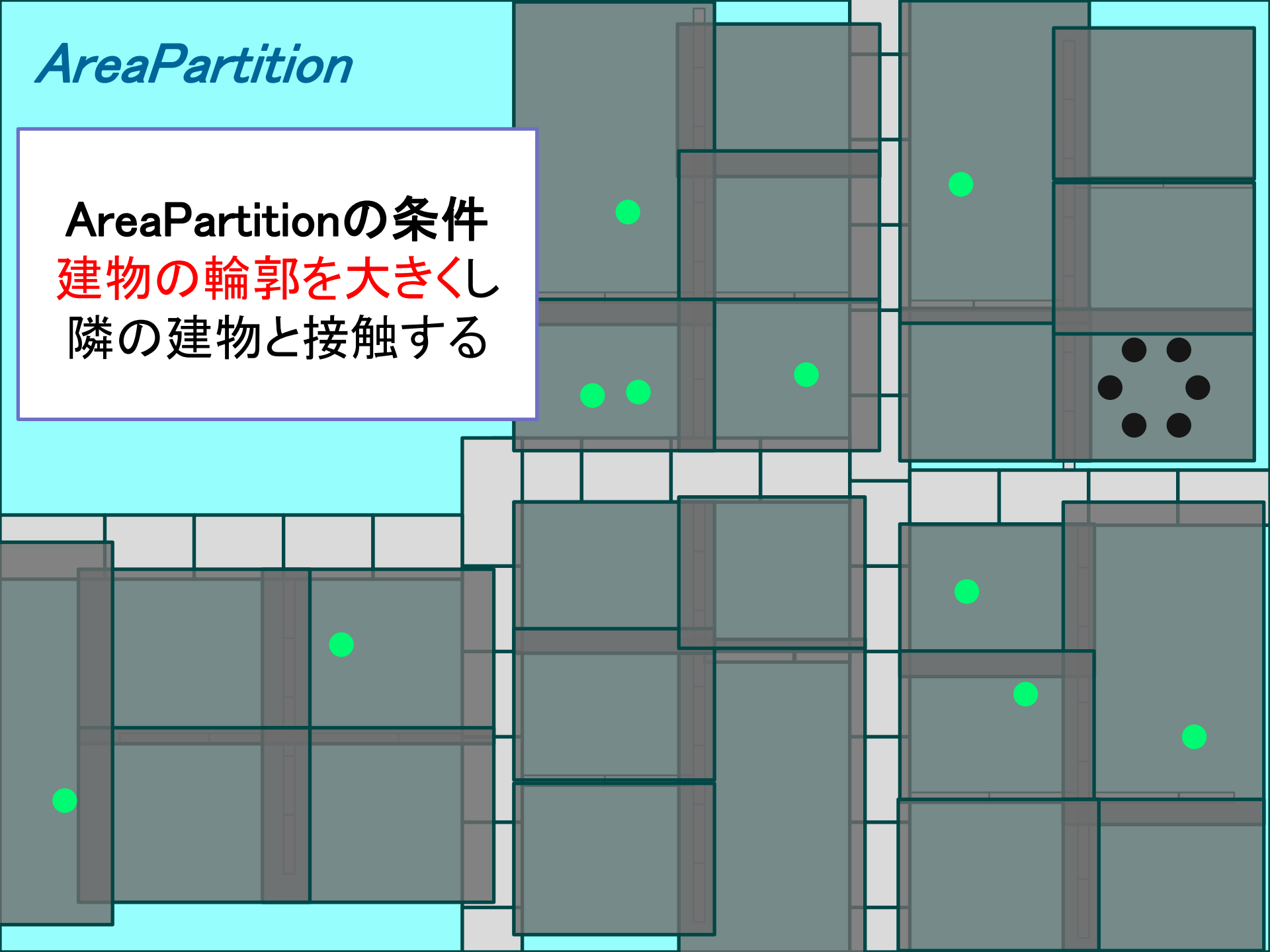
AreaPartition

AreaPartitionの条件
建物の輪郭を大きくし
隣の建物と接触する



AreaPartition

AreaPartitionの条件
建物の輪郭を大きくし
隣の建物と接触する

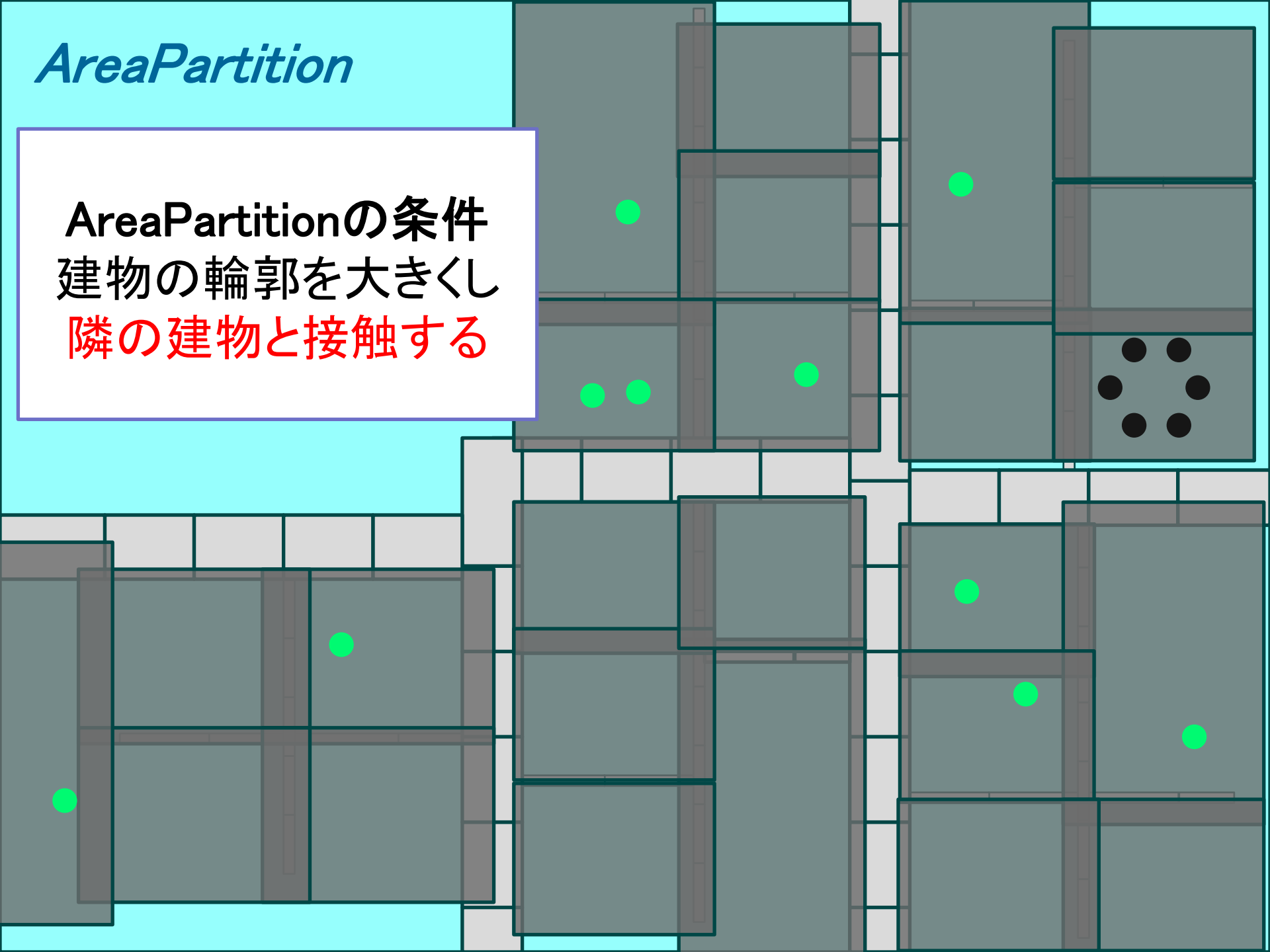


AreaPartition

AreaPartitionの条件

建物の輪郭を大きくし

隣の建物と接触する

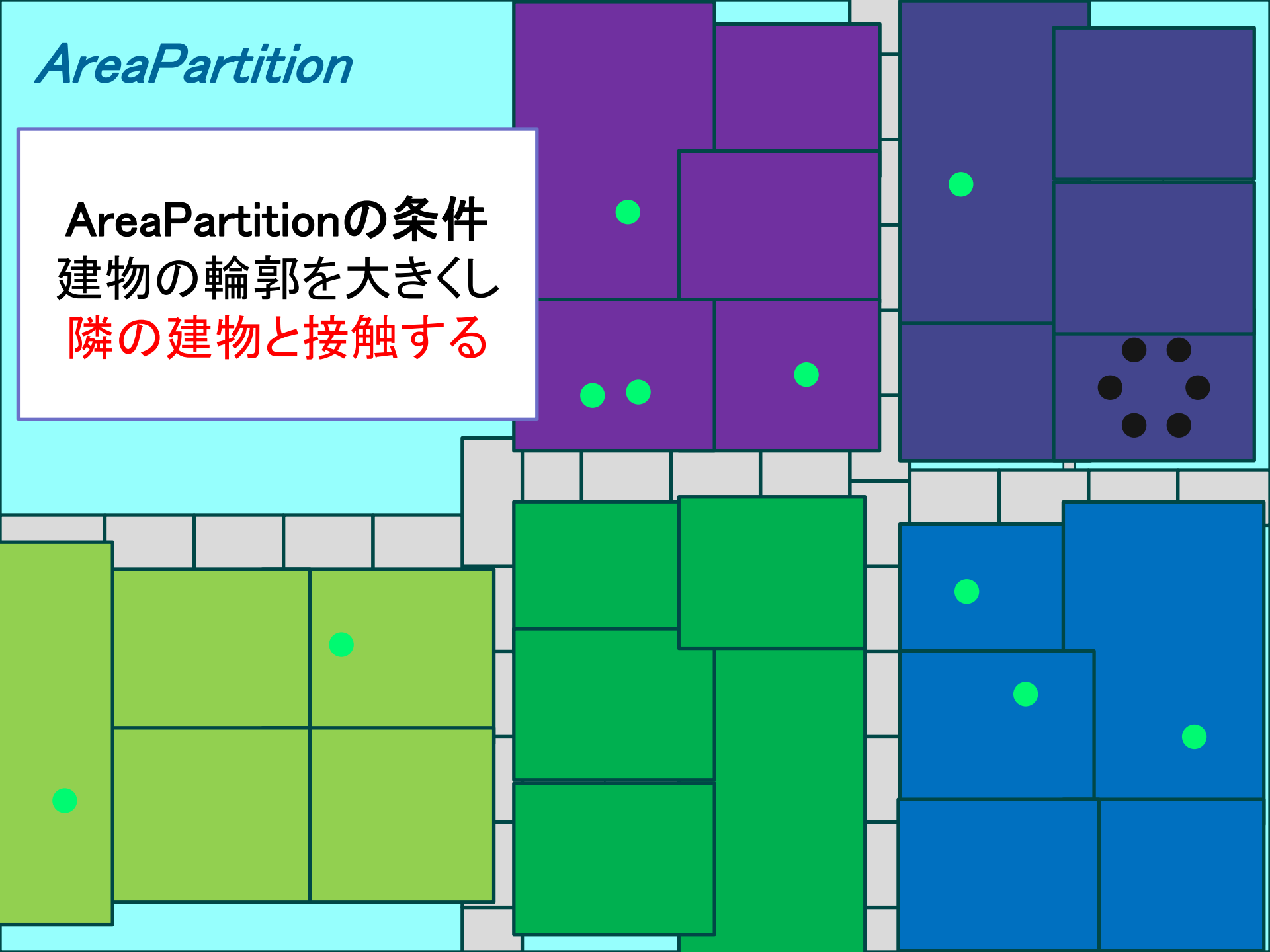


AreaPartition

AreaPartitionの条件

建物の輪郭を大きくし

隣の建物と接触する

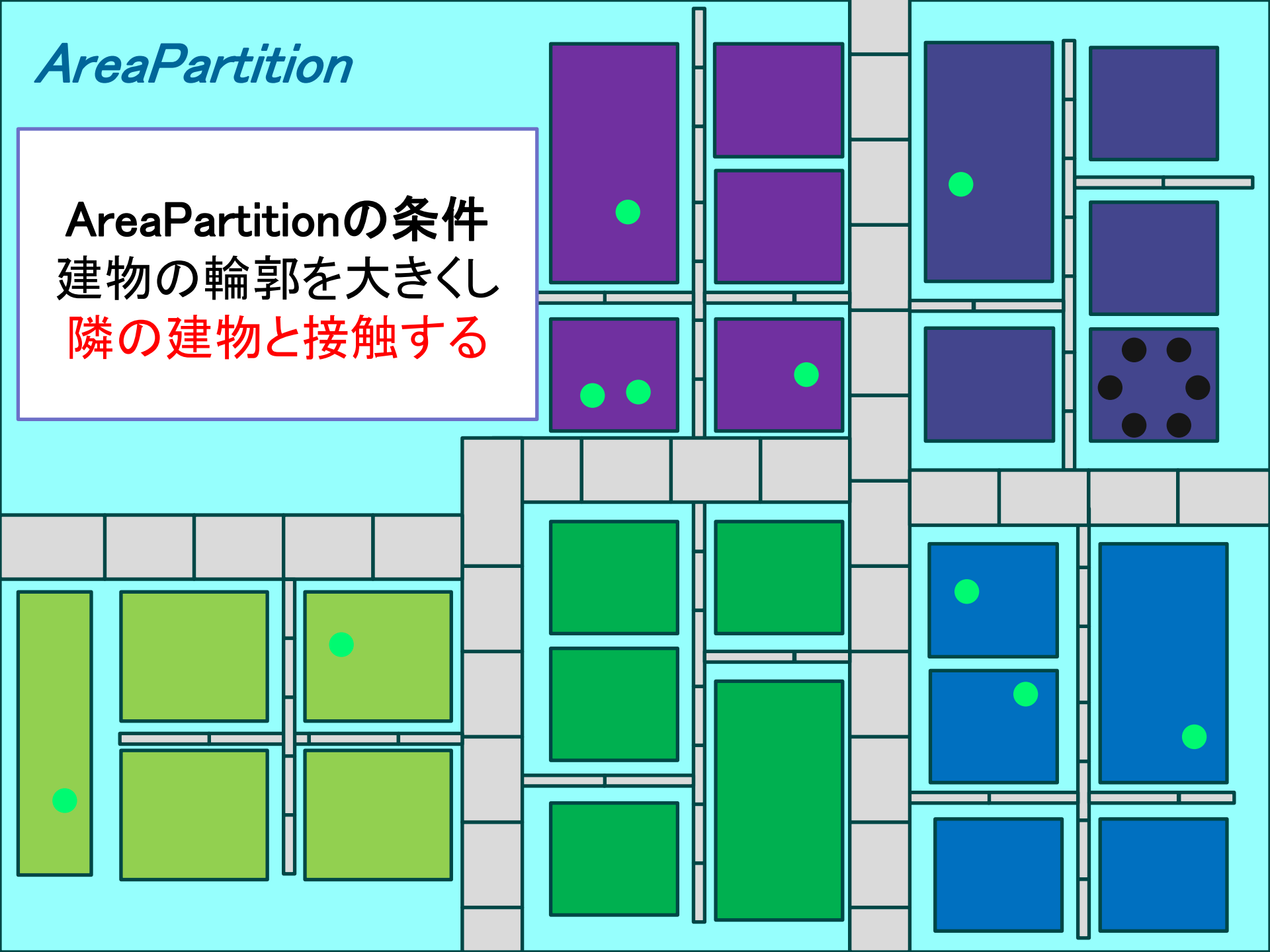


AreaPartition

AreaPartitionの条件

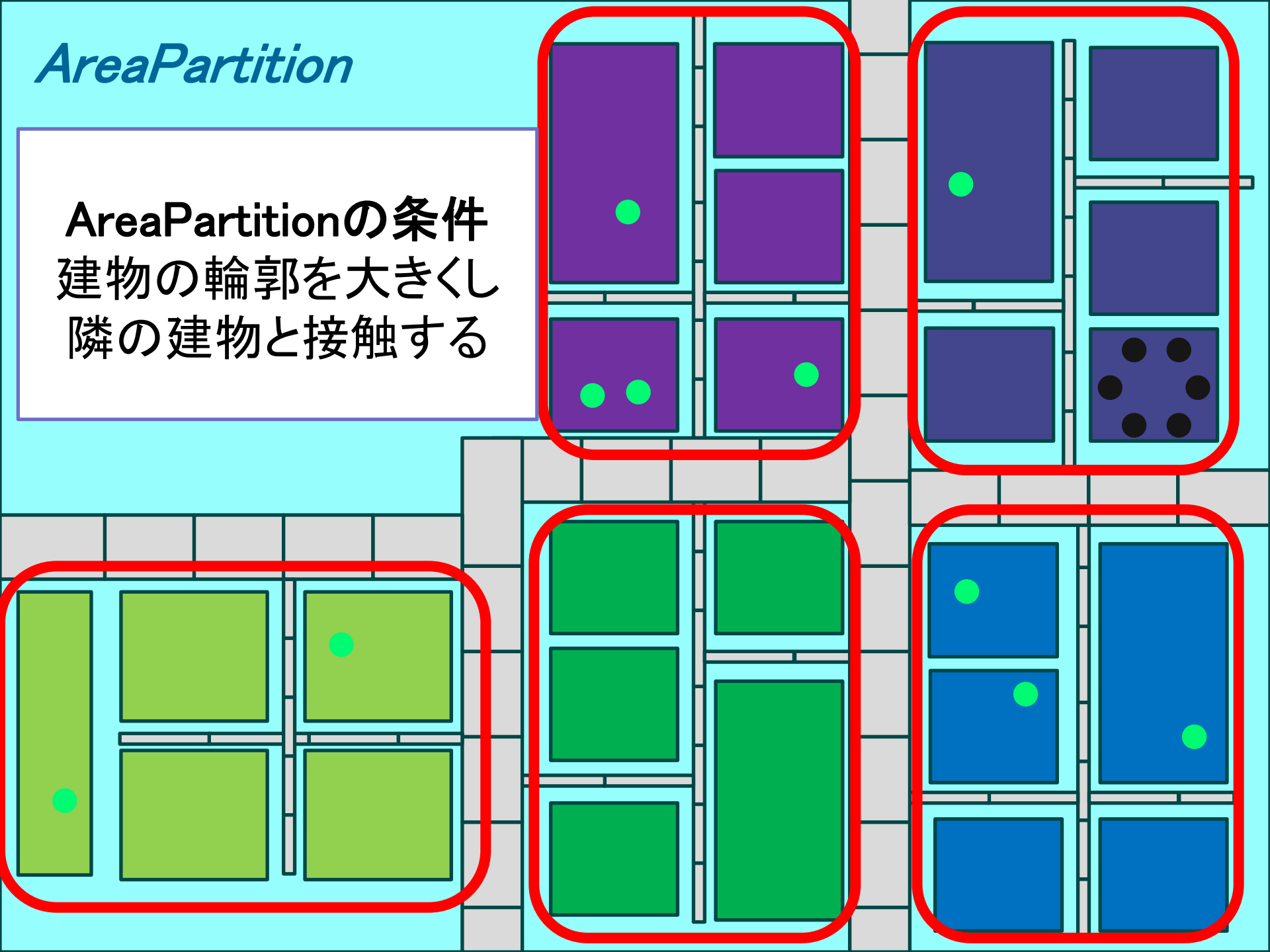
建物の輪郭を大きくし

隣の建物と接触する



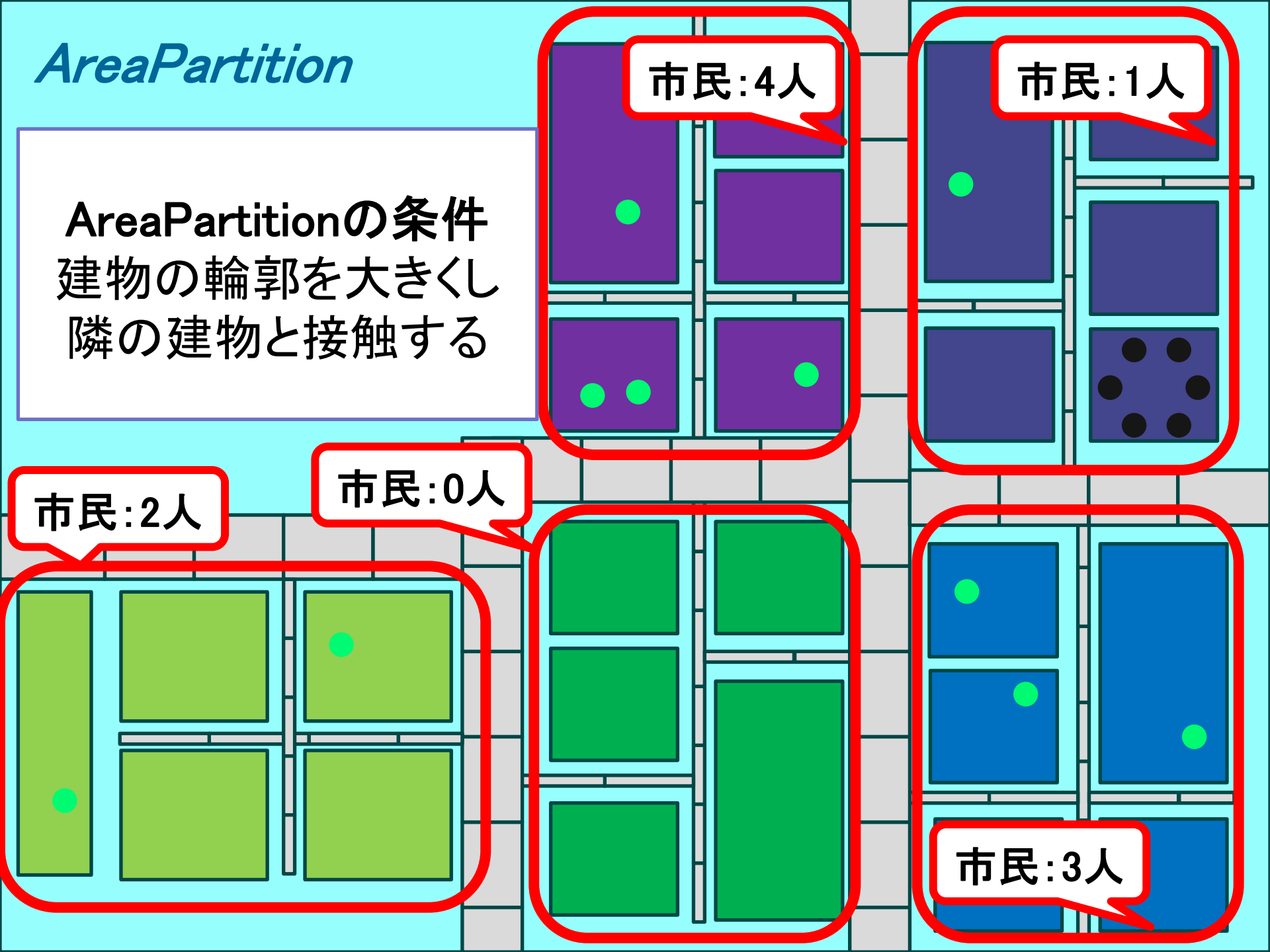
AreaPartition

AreaPartitionの条件
建物の輪郭を大きくし
隣の建物と接触する

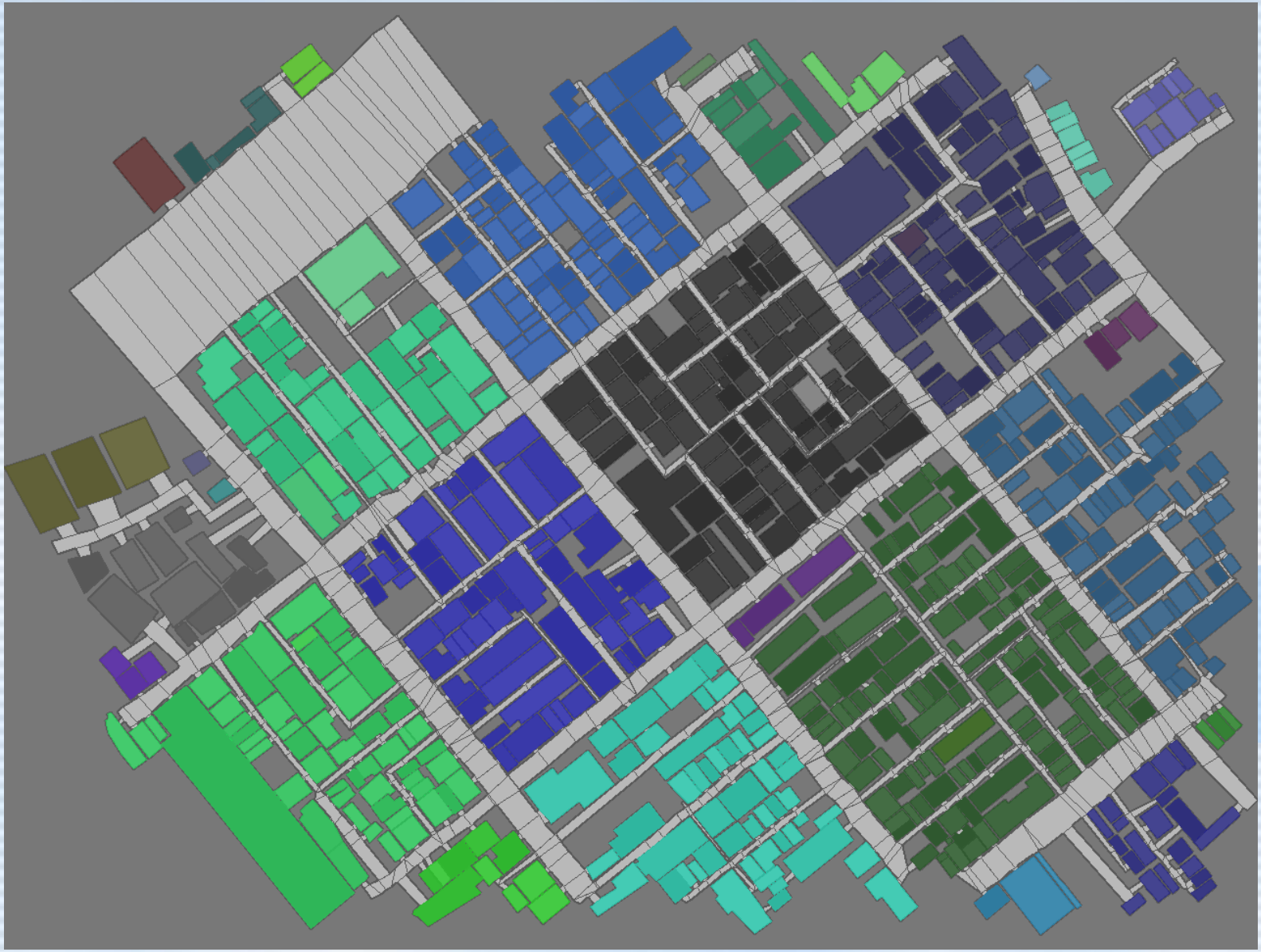


AreaPartition

AreaPartitionの条件
建物の輪郭を大きくし
隣の建物と接触する



AreaPartition



AreaPartition

結果

- 建物を区画ごとに分けATやFBの行動を効率化
- 集合で管理することで通信量を節約

課題

- 区画分けの精度を向上

アルゴリズム

- AreaPartition
- CriticalArea
- POV

CriticalArea

内容

- 到達できない交差点(CriticalArea)を定義し，PFをそこに向かわせる
- 情報が少ない時にも動ける

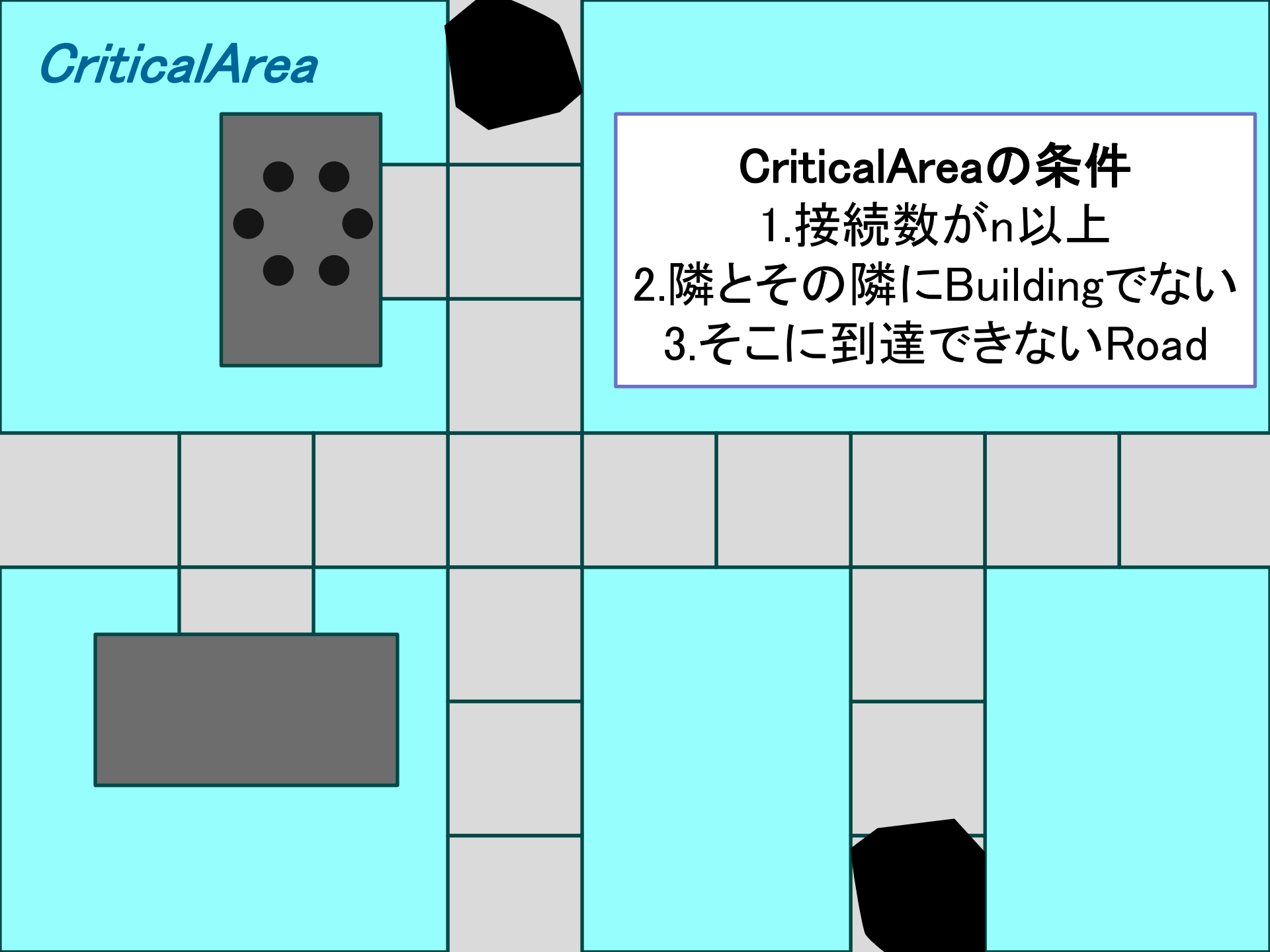
目的

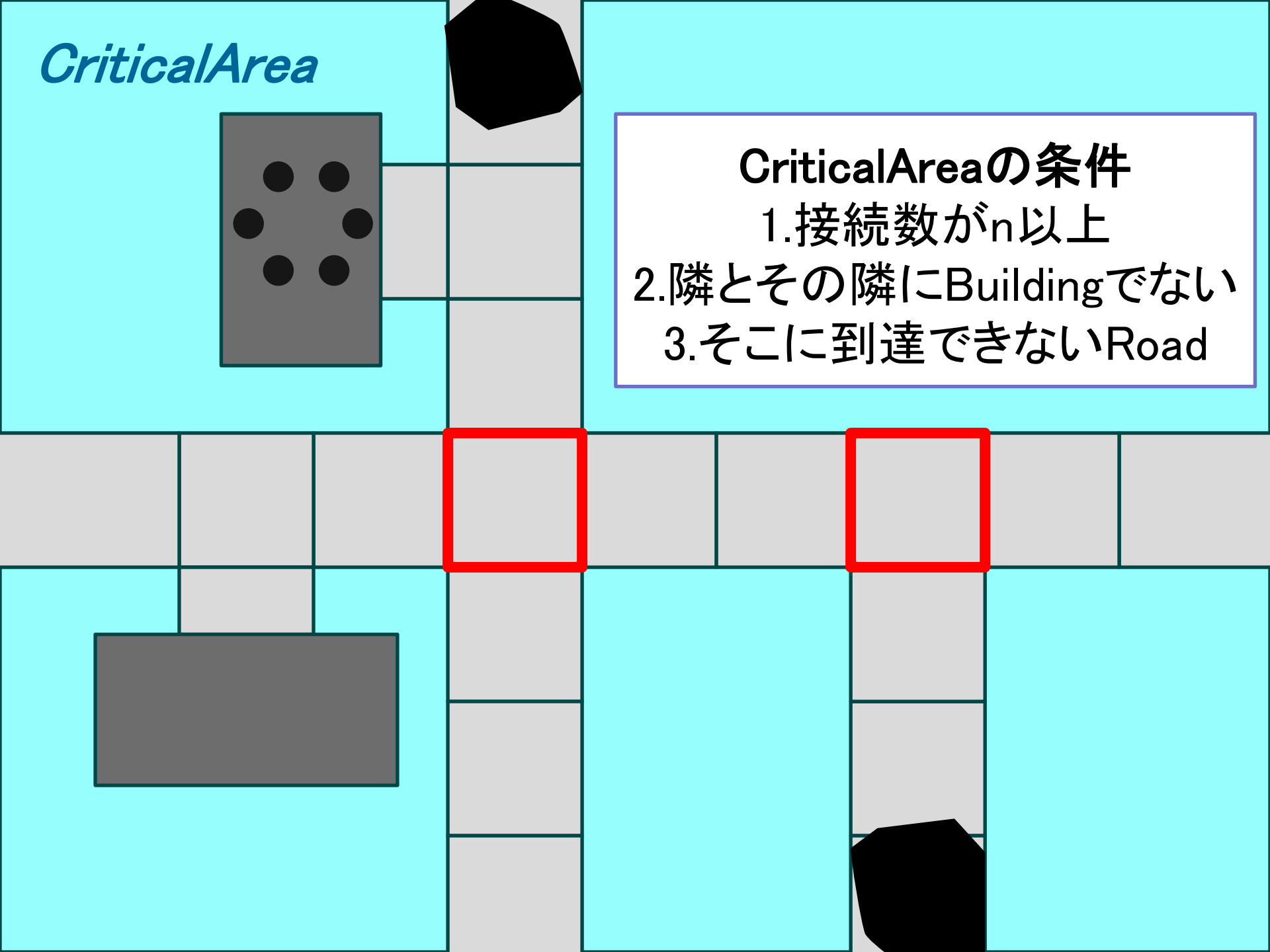
- 自分の意志だけで瓦礫除去する
- 通行可能エリアを増やす

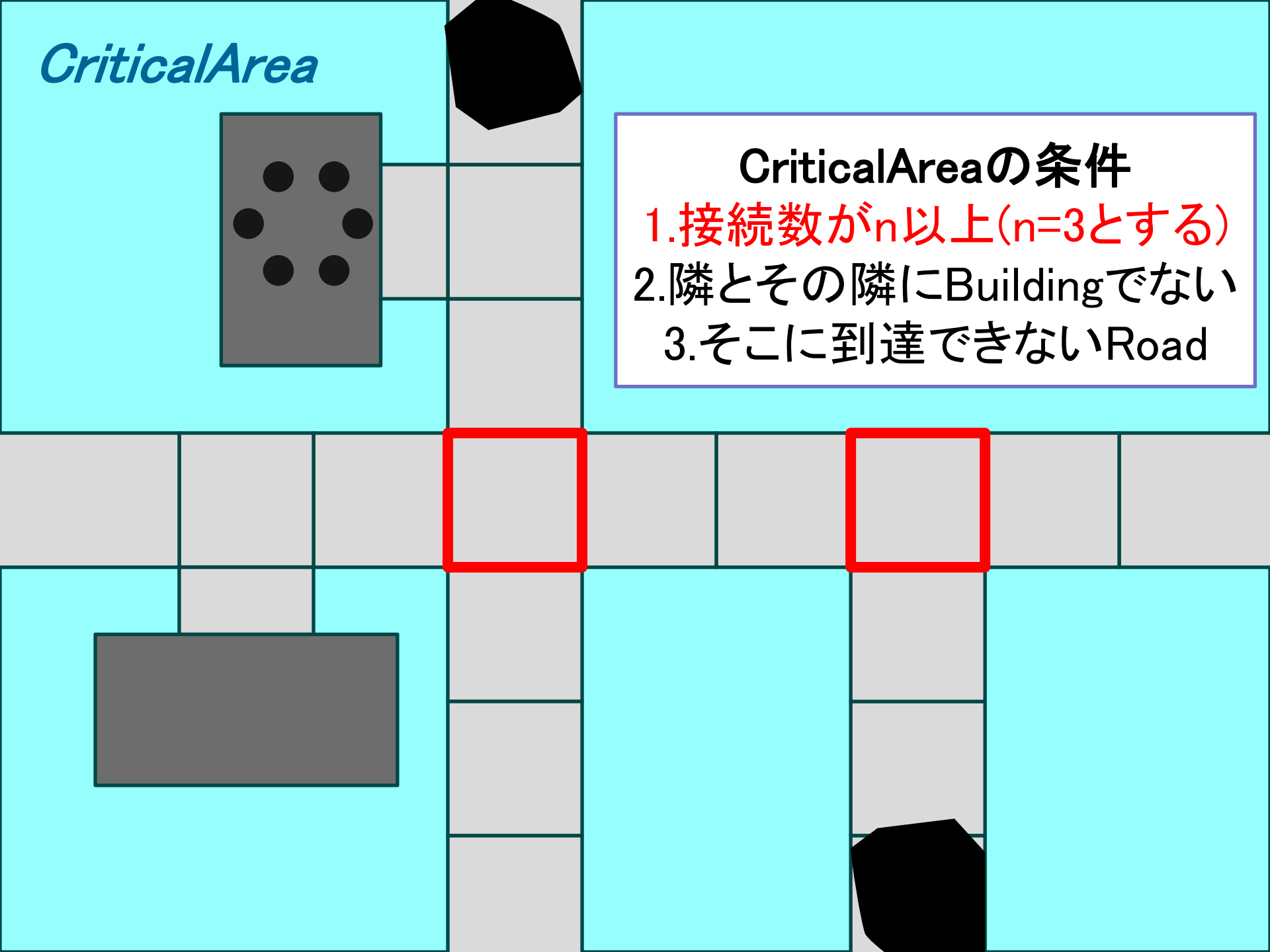
CriticalArea

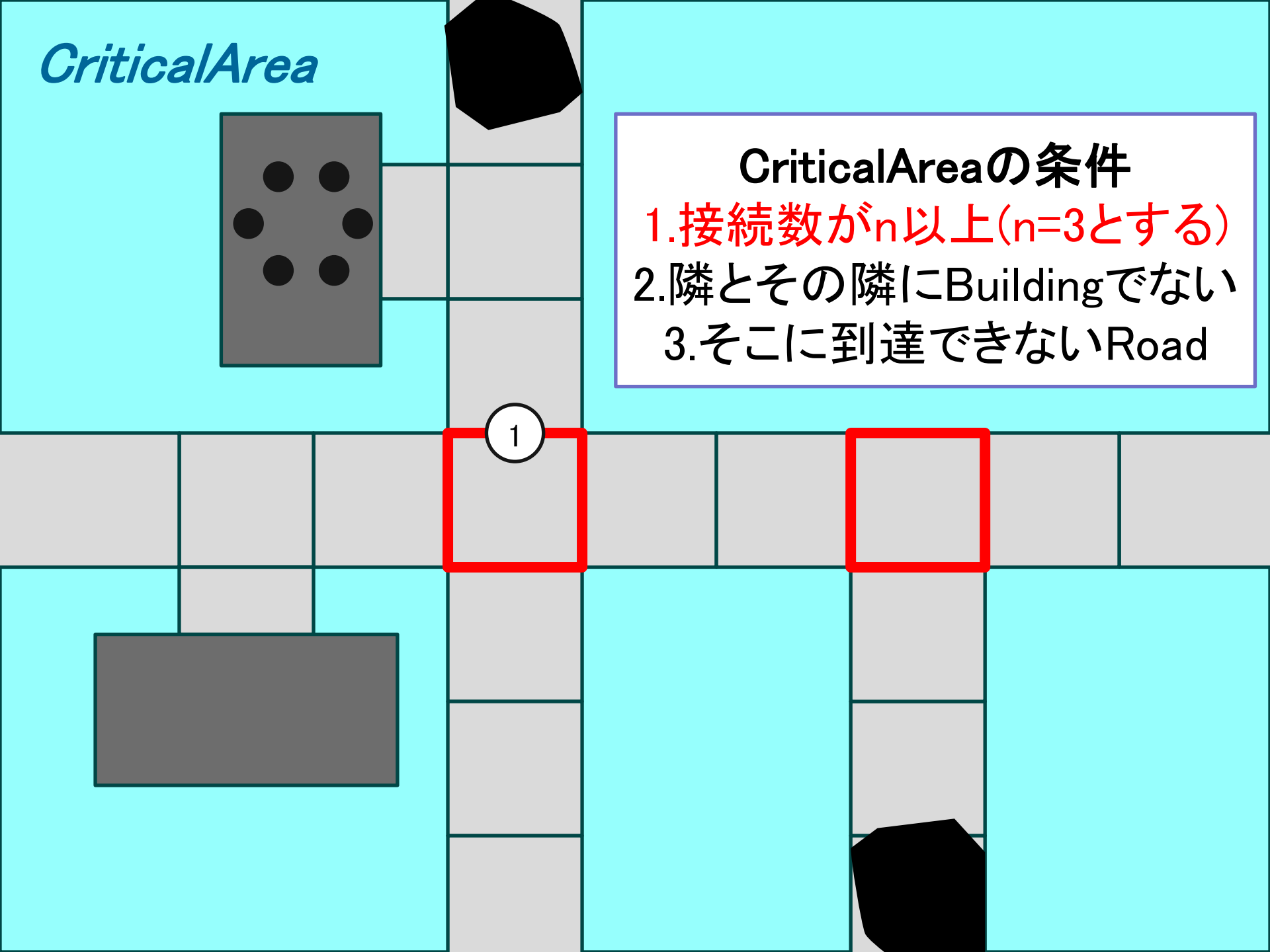
- 接続数がn以上
- 隣とその隣にBuildingが存在しない
- そこに到達できないRoad

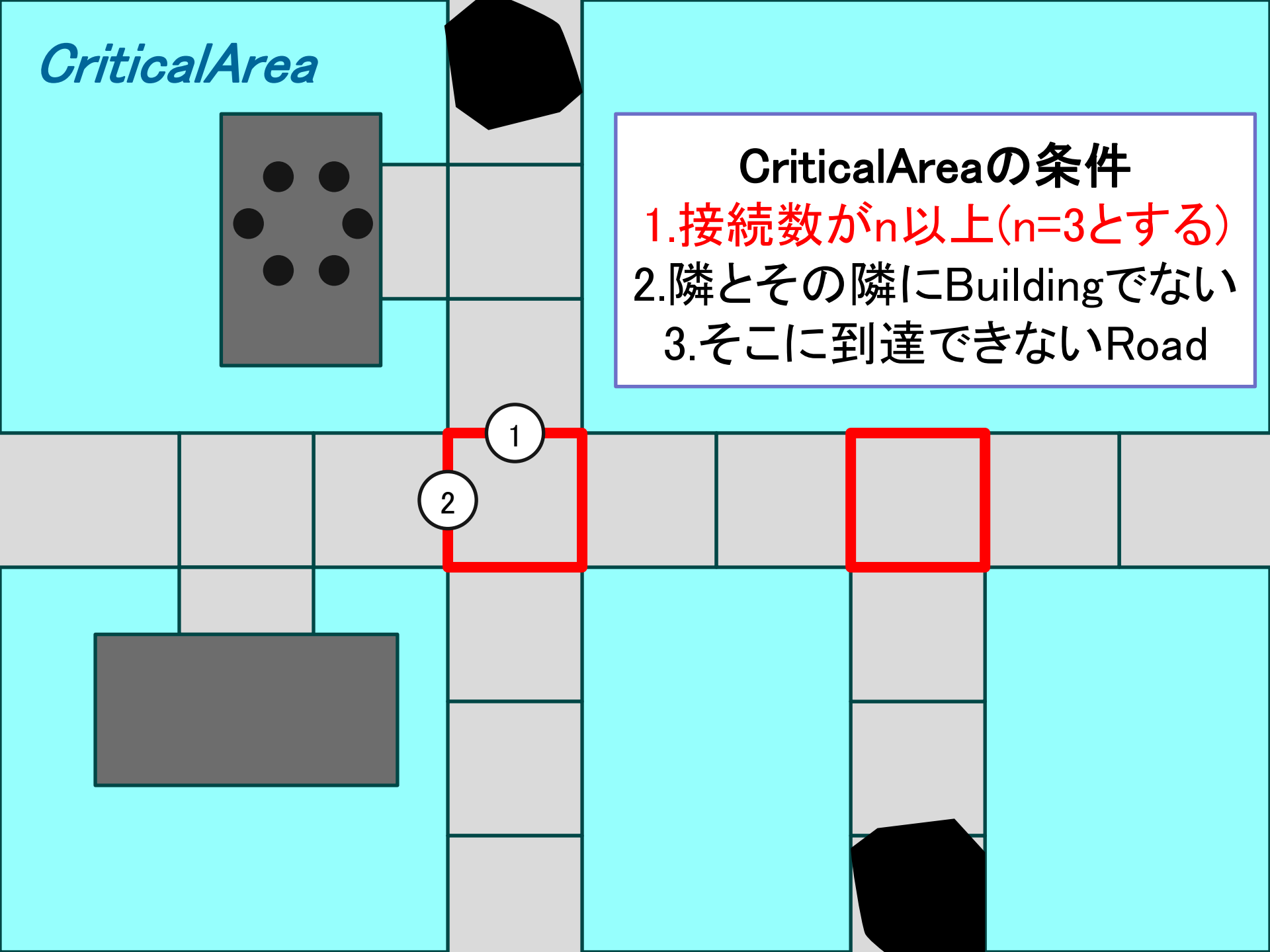
交差点を
イメージ

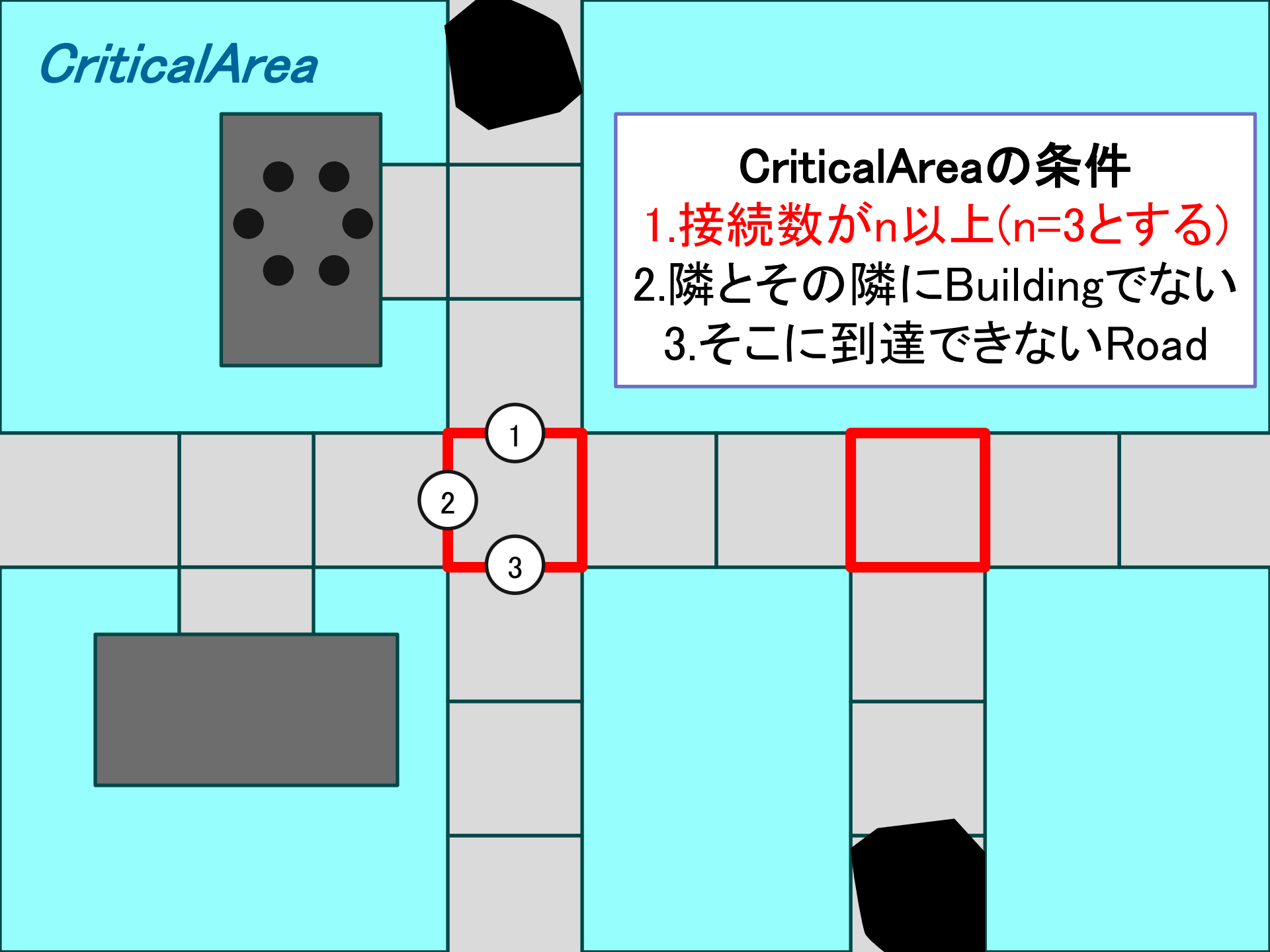

CriticalArea
CriticalAreaの条件
1.接続数がn以上
2.隣とその隣にBuildingでない
3.そこに到達できないRoad

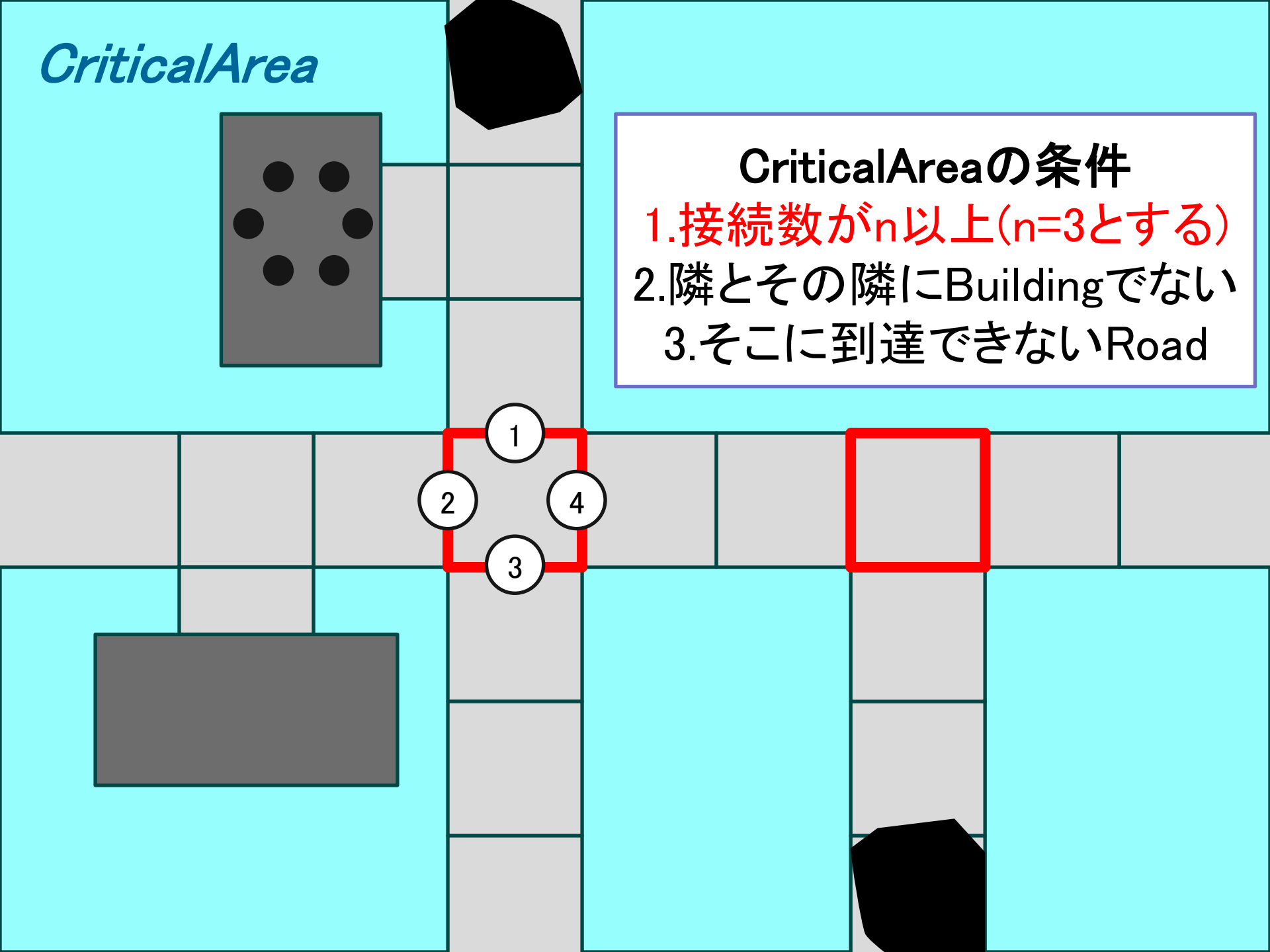

CriticalArea
CriticalAreaの条件
1.接続数がn以上
2.隣とその隣にBuildingでない
3.そこに到達できないRoad


CriticalArea
CriticalAreaの条件
1.接続数がn以上(n=3とする)
2.隣とその隣にBuildingでない
3.そこに到達できないRoad

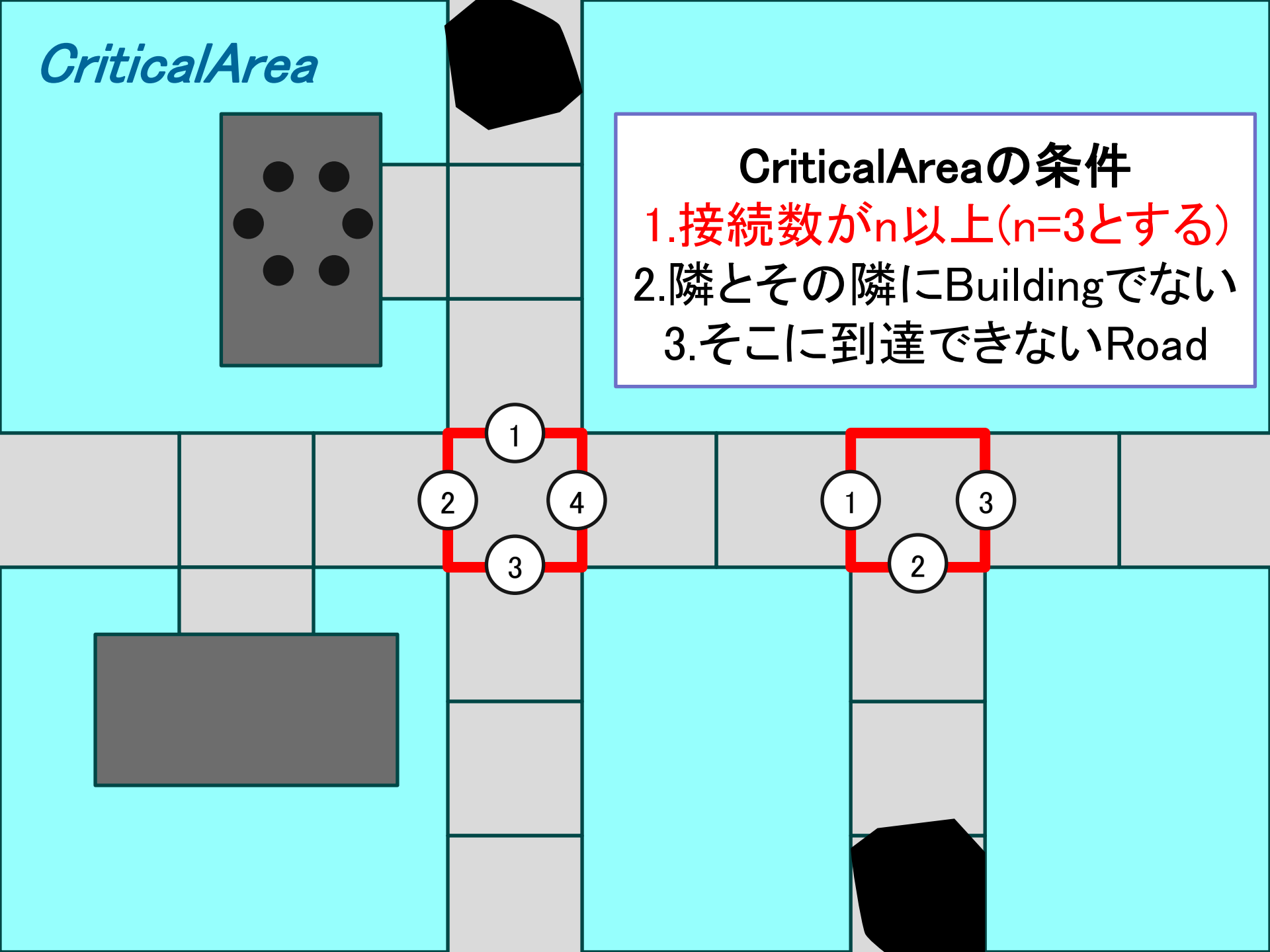

CriticalArea
CriticalAreaの条件
1.接続数がn以上(n=3とする)
2.隣とその隣にBuildingでない
3.そこに到達できないRoad
1

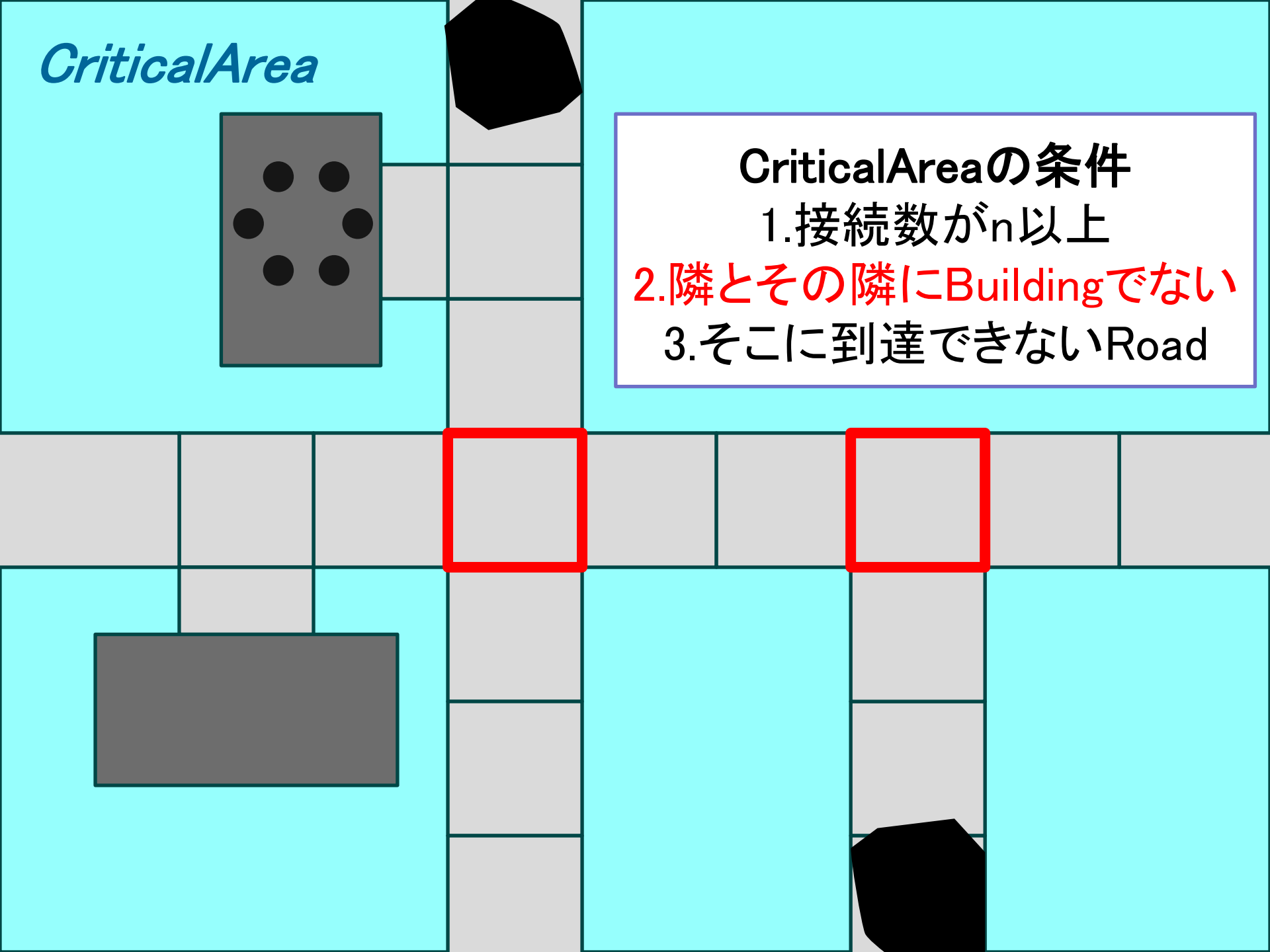

CriticalArea
CriticalAreaの条件
1.接続数がn以上(n=3とする)
2.隣とその隣にBuildingでない
3.そこに到達できないRoad
1
2

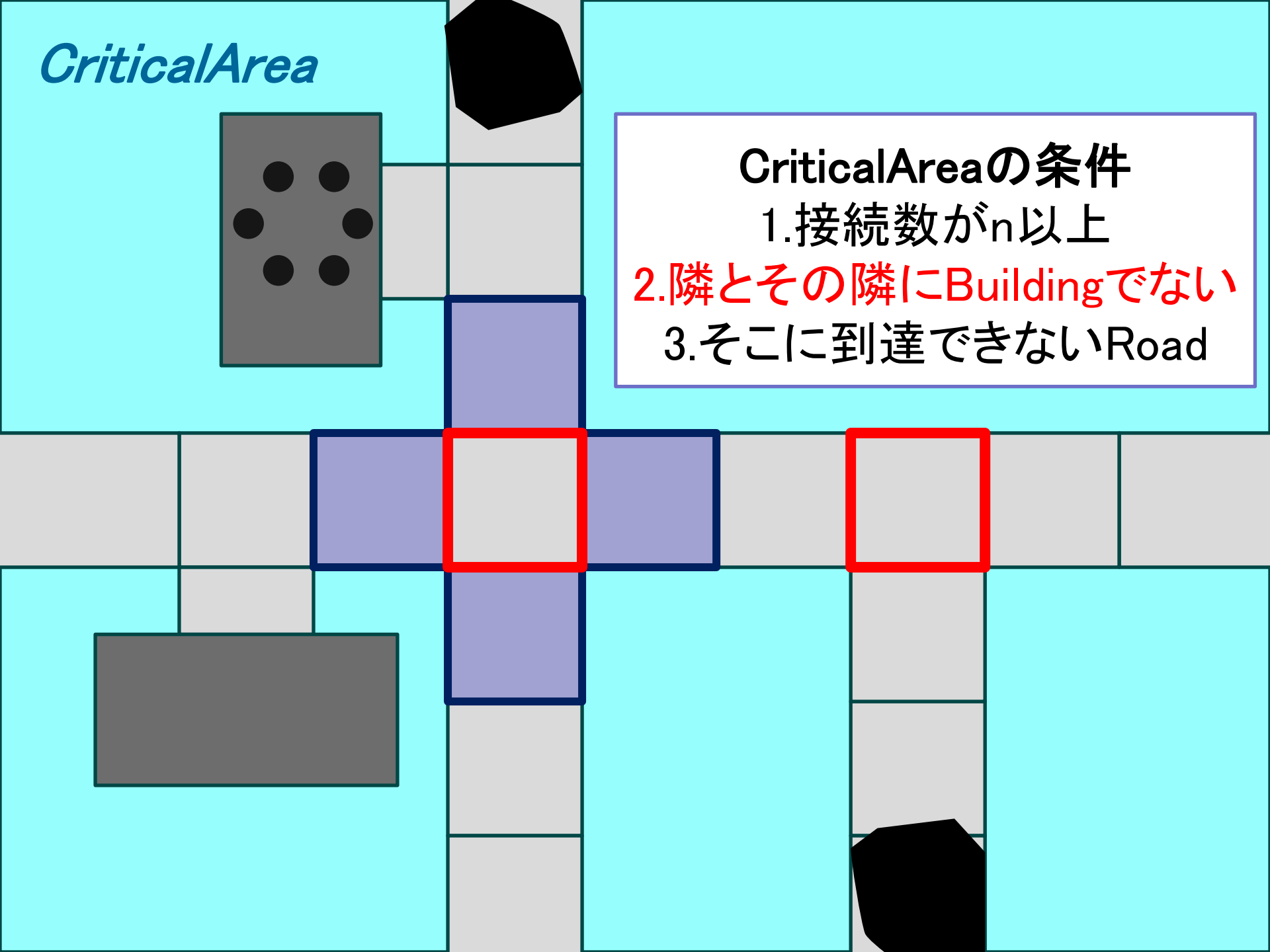

CriticalArea
CriticalAreaの条件
1.接続数がn以上(n=3とする)
2.隣とその隣にBuildingでない
3.そこに到達できないRoad
1
2
3


CriticalArea
CriticalAreaの条件
1.接続数がn以上(n=3とする)
2.隣とその隣にBuildingでない
3.そこに到達できないRoad
1
2
3
4

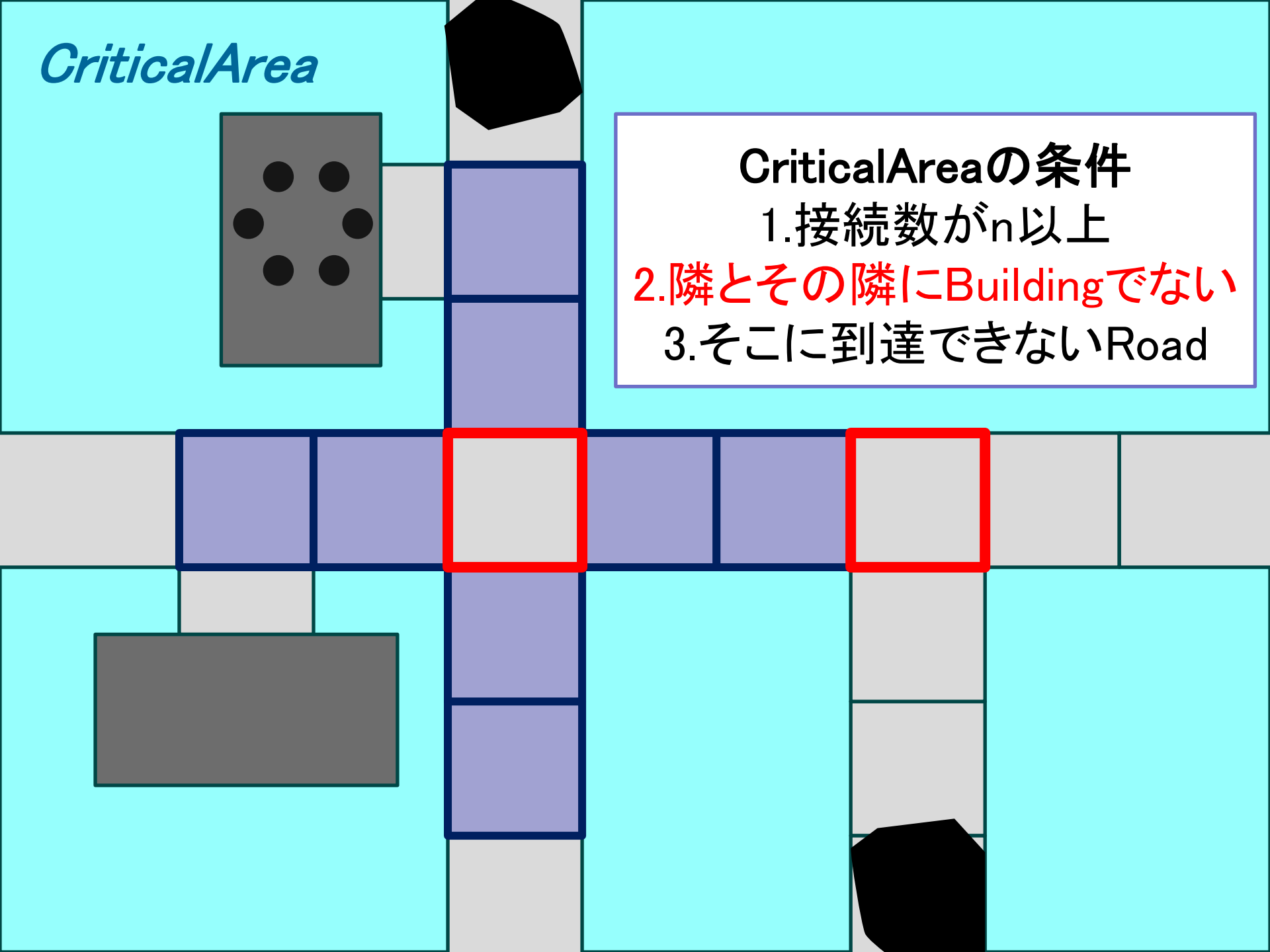
CriticalArea




CriticalArea
CriticalAreaの条件
1.接続数がn以上
2.隣とその隣にBuildingでない
3.そこに到達できないRoad


CriticalArea
CriticalAreaの条件
1.接続数がn以上
2.隣とその隣にBuildingでない
3.そこに到達できないRoad

CriticalArea

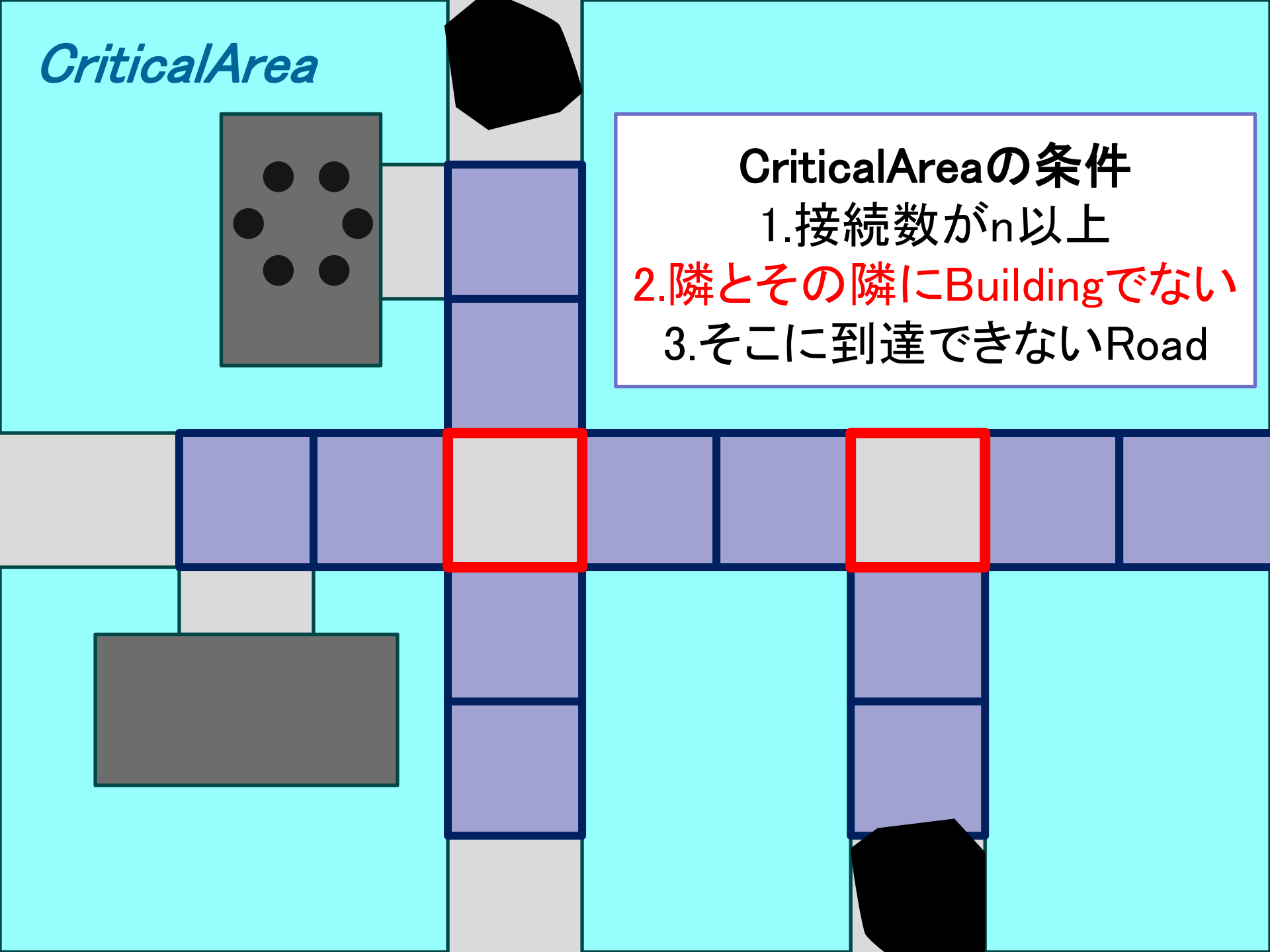


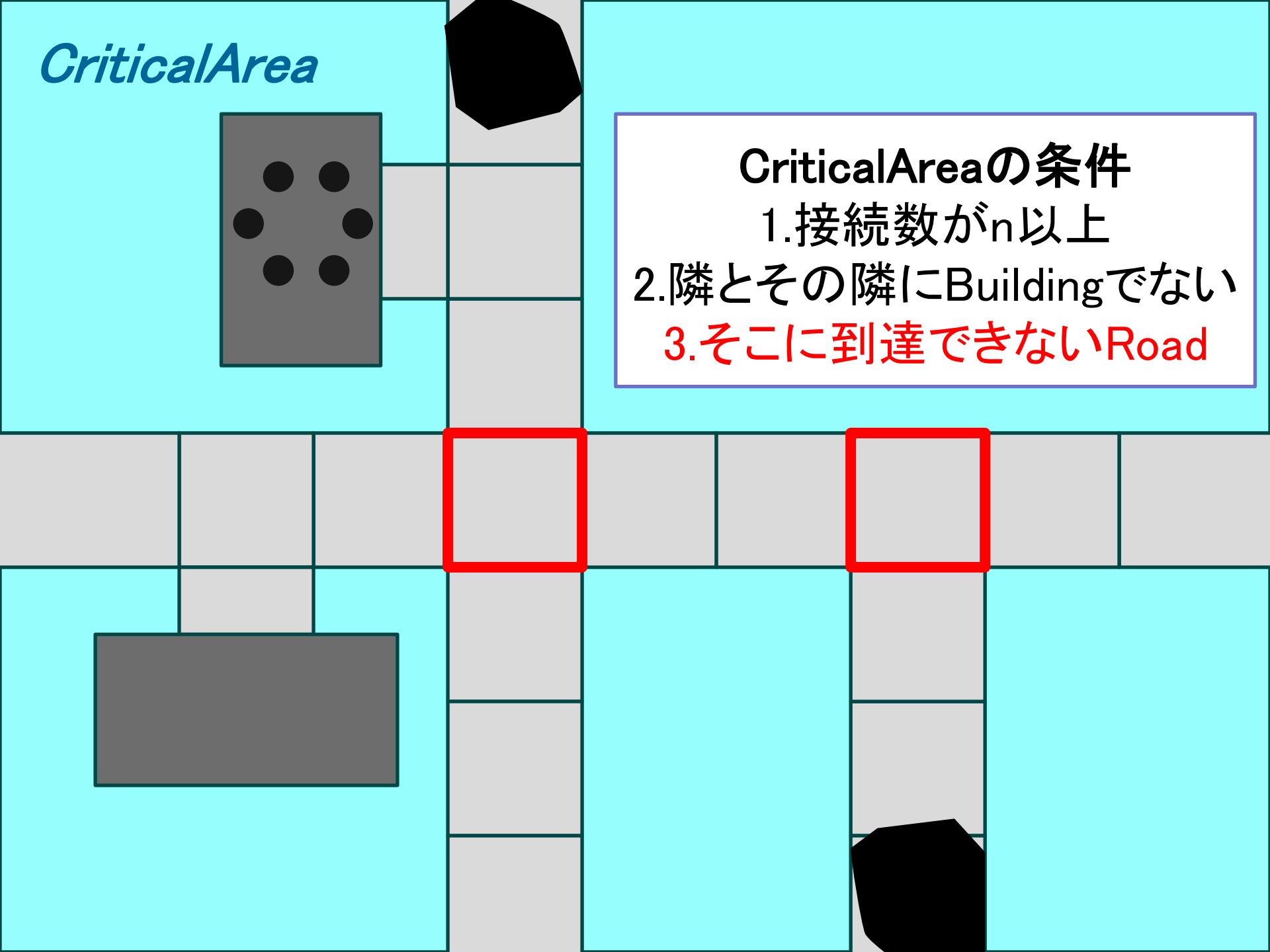
CriticalAreaの条件

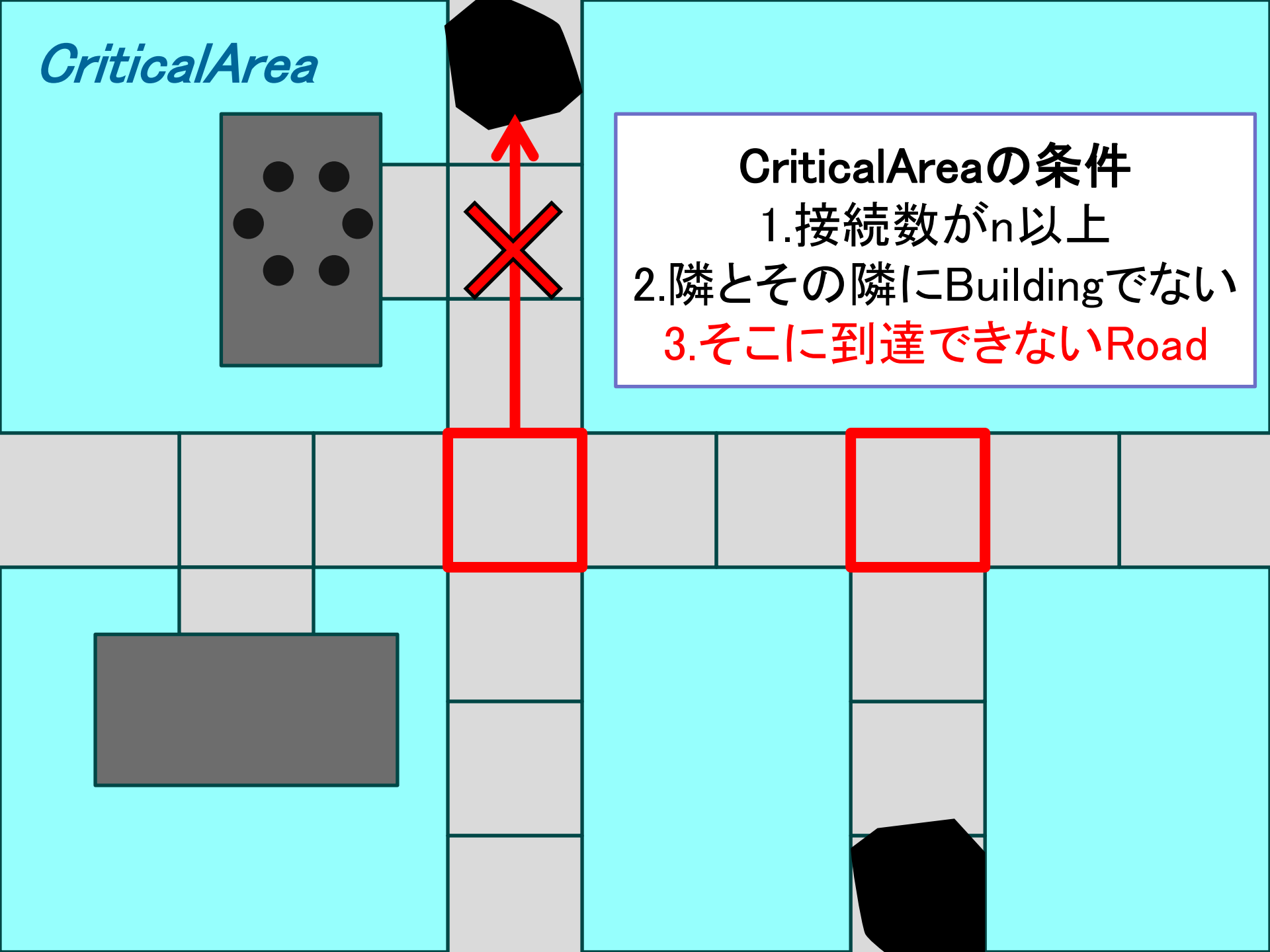
1.接続数がn以上

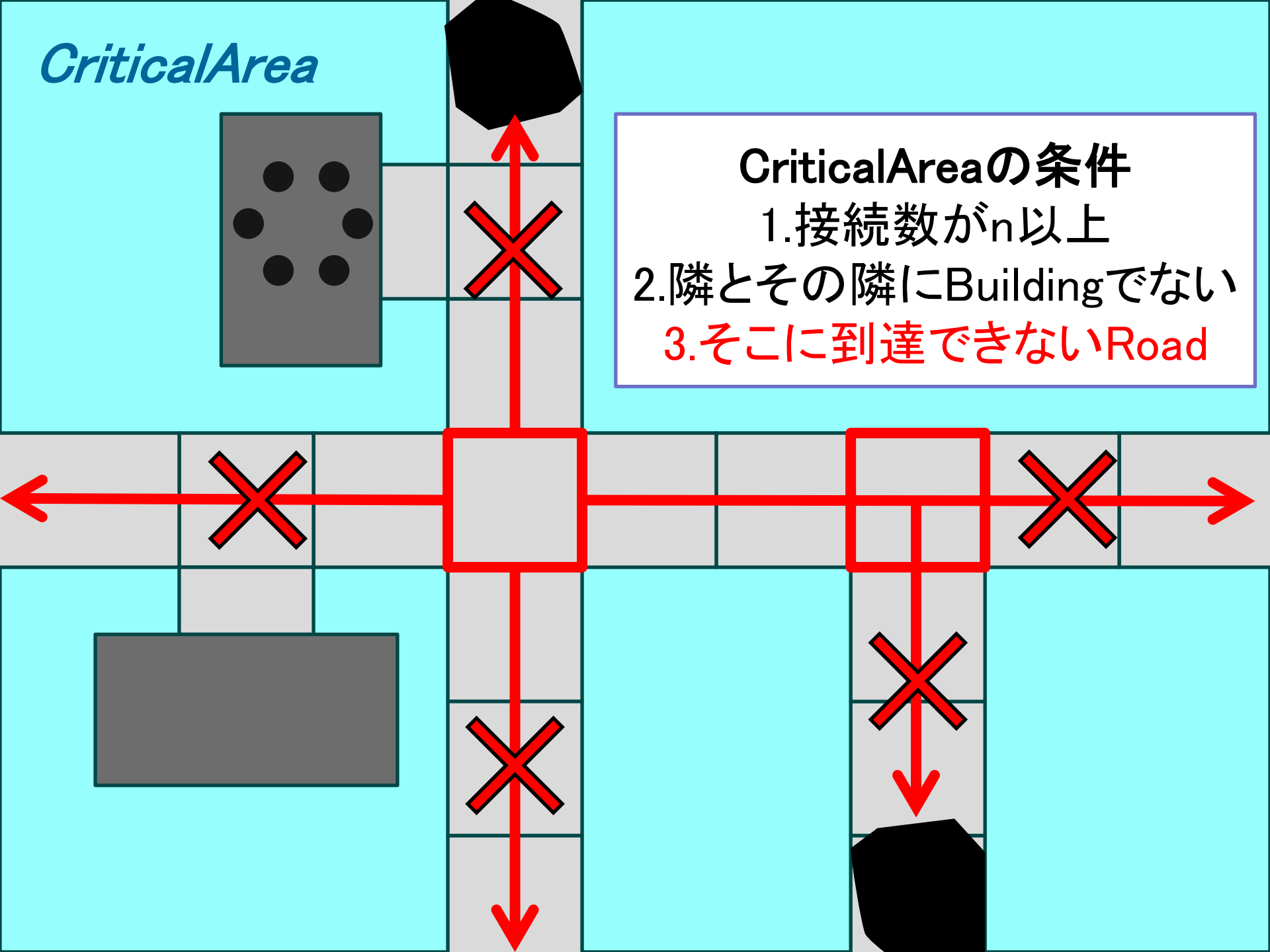
2.隣とその隣にBuildingでない

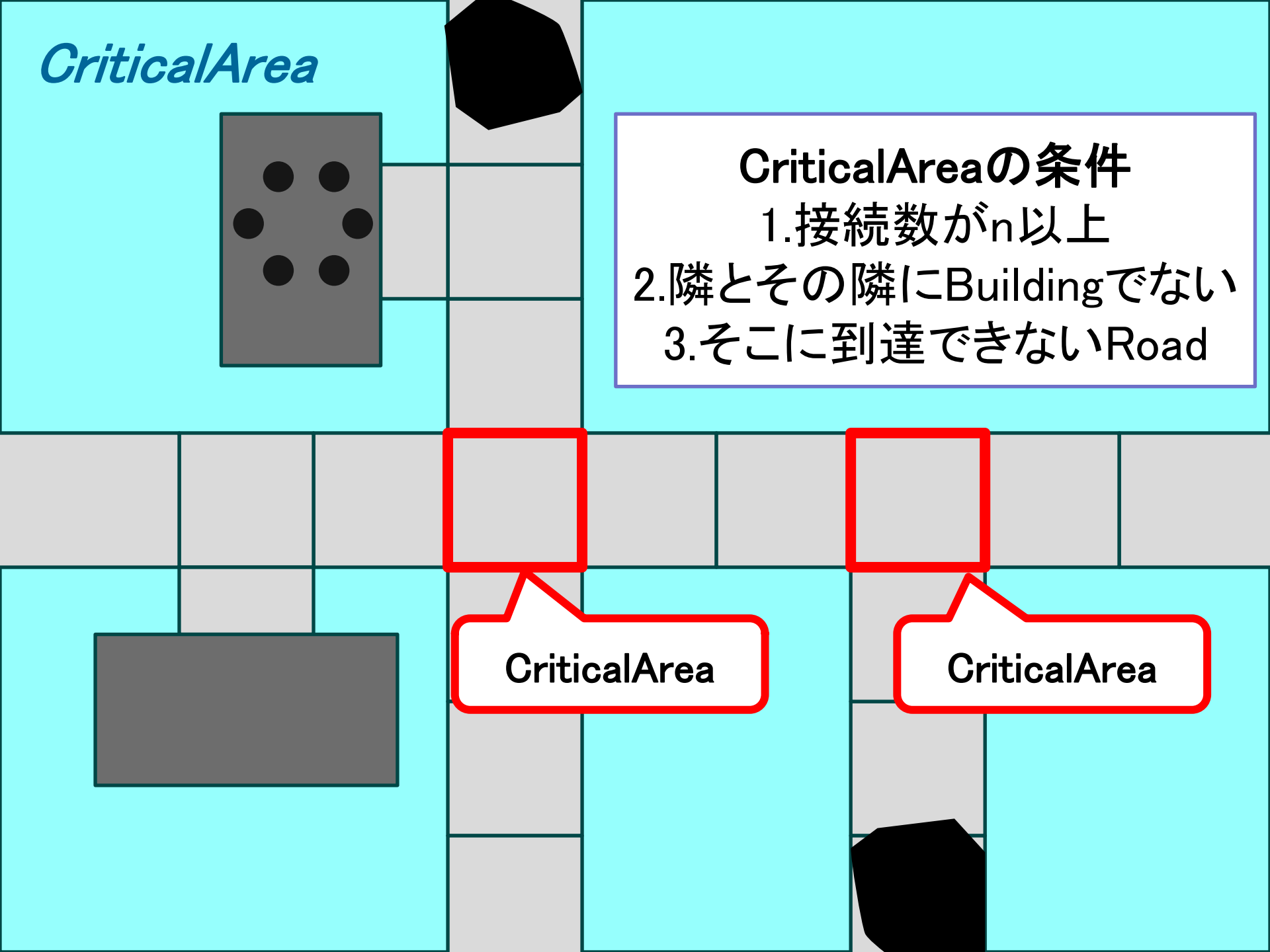
3.そこに到達できないRoad


CriticalArea
CriticalAreaの条件
1.接続数がn以上
2.隣とその隣にBuildingでない
3.そこに到達できないRoad

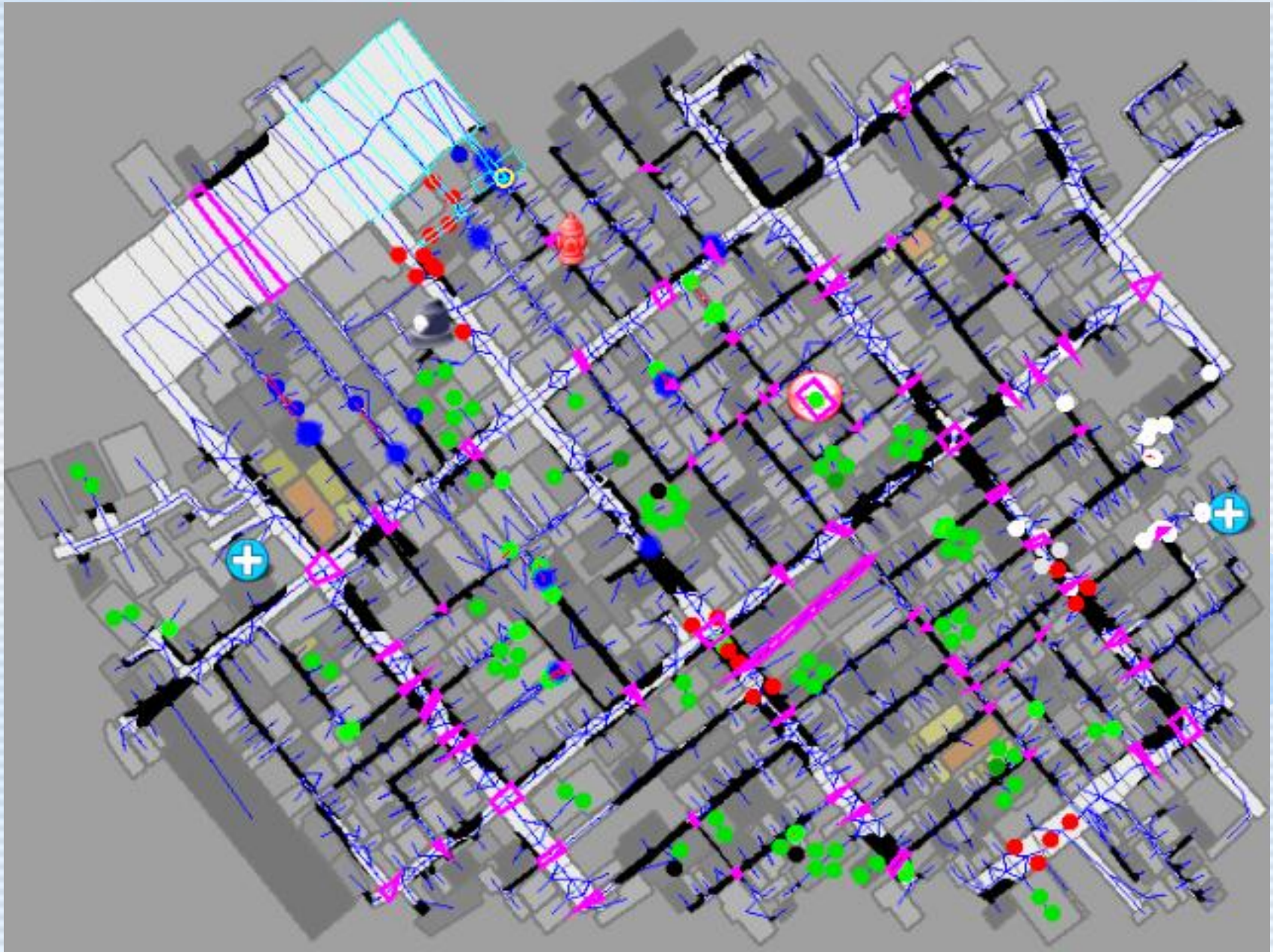

CriticalArea
CriticalAreaの条件
1.接続数がn以上
2.隣とその隣にBuildingでない
3.そこに到達できないRoad


CriticalArea
CriticalAreaの条件
1.接続数がn以上
2.隣とその隣にBuildingでない
3.そこに到達できないRoad


CriticalArea
CriticalAreaの条件
1.接続数がn以上
2.隣とその隣にBuildingでない
3.そこに到達できないRoad


CriticalArea
CriticalAreaの条件
1.接続数がn以上
2.隣とその隣にBuildingでない
3.そこに到達できないRoad
CriticalArea
CriticalArea

CriticalArea



CriticalArea

結果

- 全ての交差点を繋ぐことが出来る
- 必要な情報が少ないので，自身で考えて動ける

課題

- マップによっては交差点が多すぎる
- 通行可能ならどんなに遠回りでもCritical Areaではない

CriticalArea

結果

- 全ての交差点を繋ぐことが出来る
- 必要な情報が少ないので，自身で考えて動ける

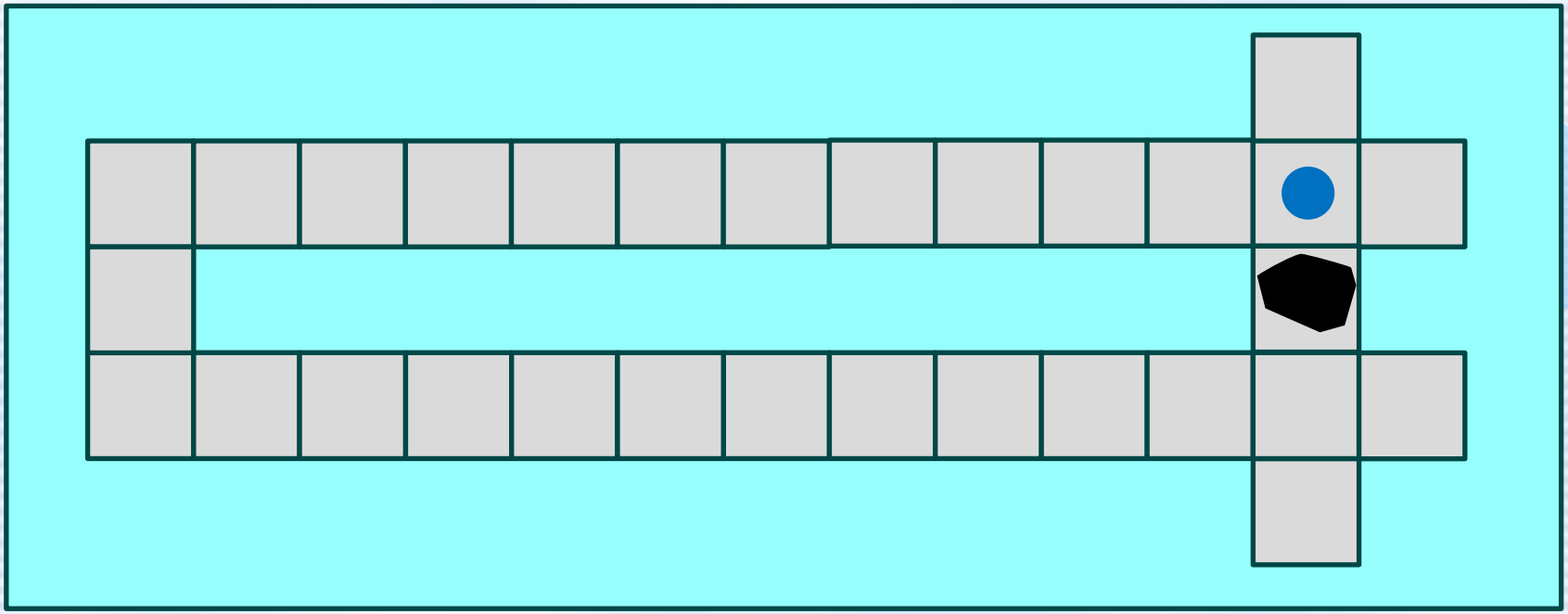
課題

- マップによっては交差点が多すぎる
- 通行可能ならどんなに遠回りでもCritical Areaではない

CriticalArea

課題

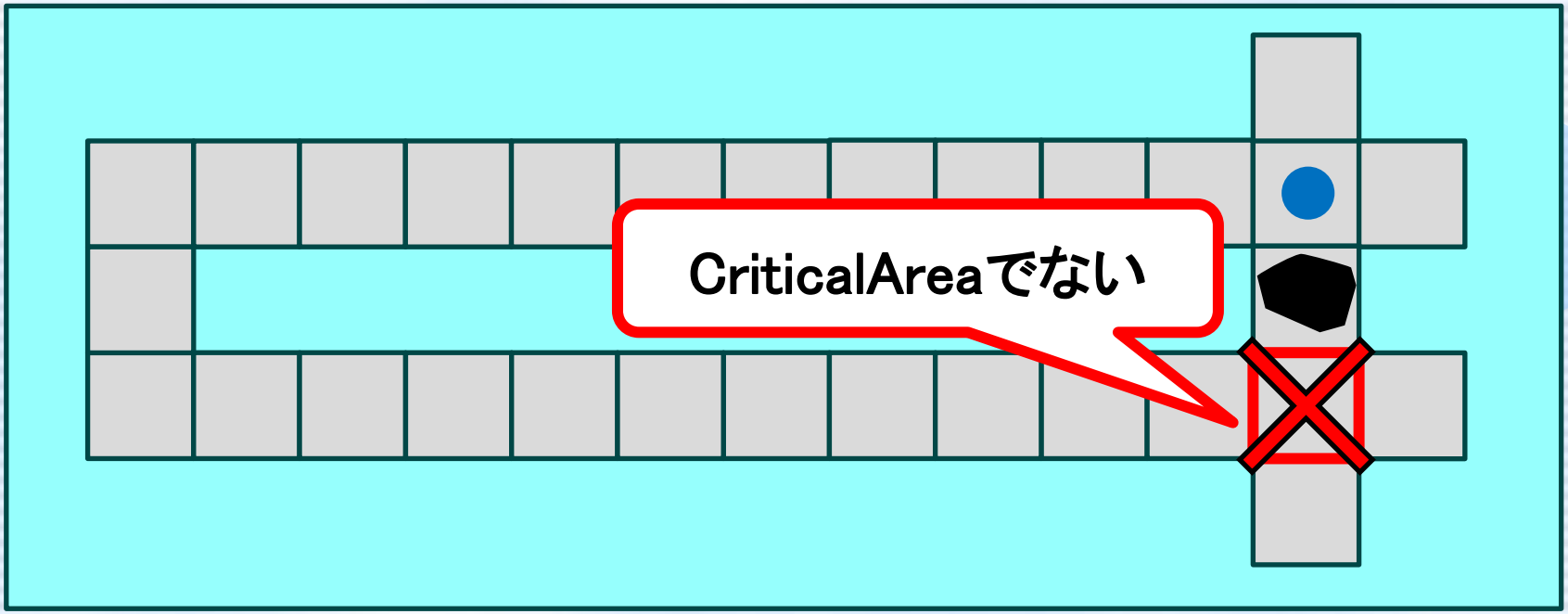
- マップによっては交差点が多すぎる
- 通行可能ならどんなに遠回りでもCritical Areaではない



CriticalArea

課題

- マップによっては交差点が多すぎる
- 通行可能ならどんなに遠回りでもCritical Areaではない



アルゴリズム

- AreaPartition
- CriticalArea
- POV

POV

内容

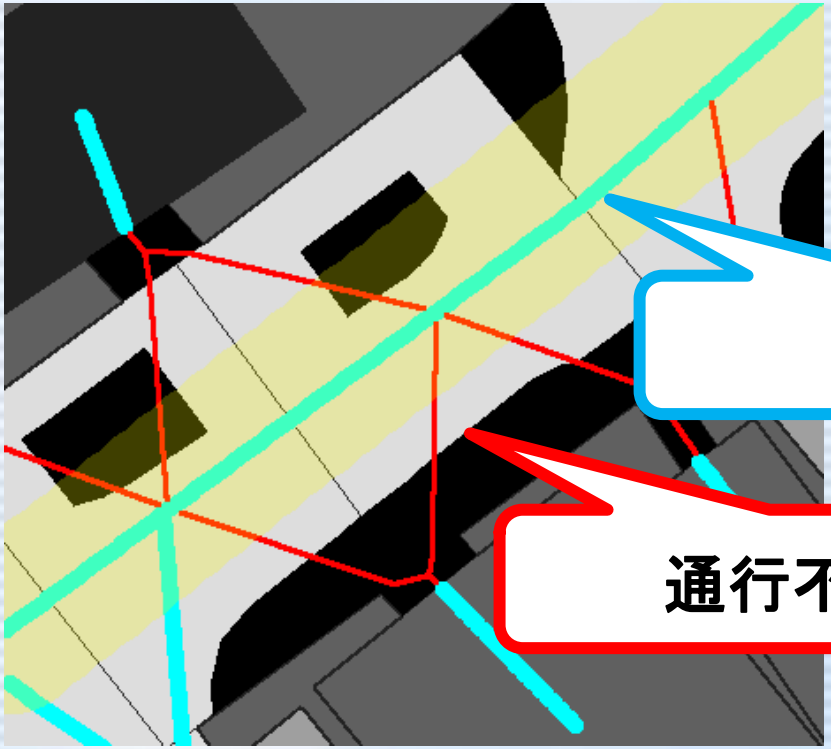
・始点と終点までの隣接する各ノードを見ていき，通行可能か判断する

目的

・的確な通行可能判断

POV

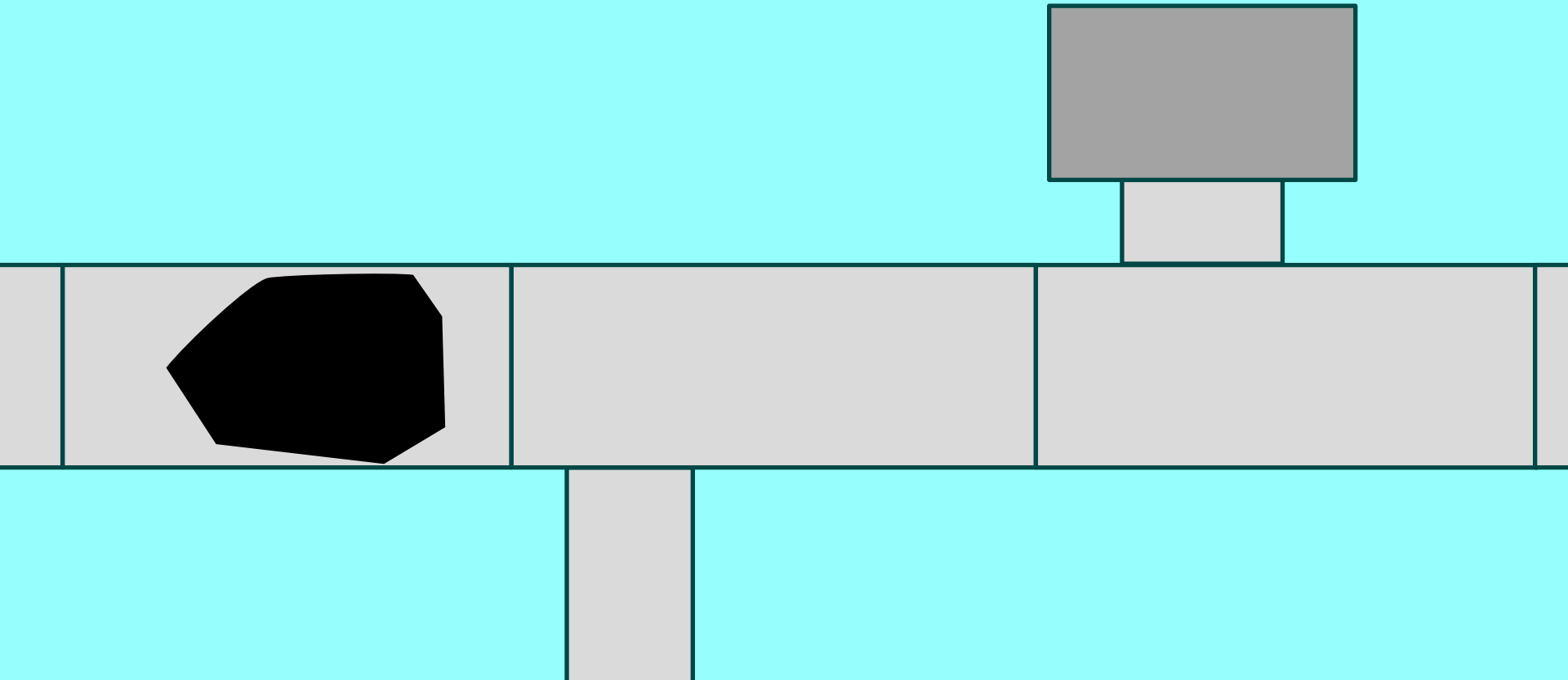
- Roadの隣接部分の短い方の中点と中心を線で繋ぐ
- その線と瓦礫が接したら通行不可



POV（通行可能判断）

POVの条件

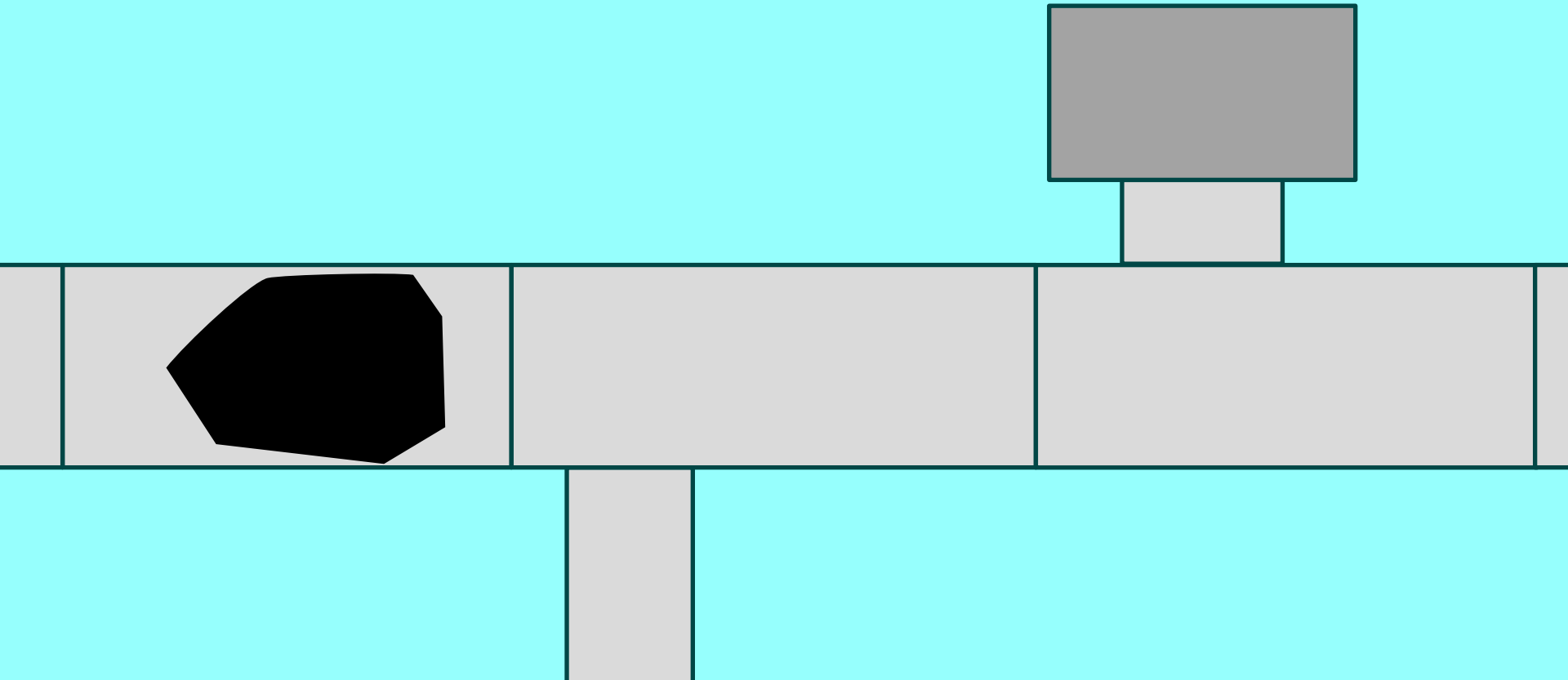
- Roadの隣接部分の線分短い方の中点とRoadとBuildingの中心を線で繋ぐ
- その線と瓦礫が接したら通行不可



POV（通行可能判断）

POVの条件

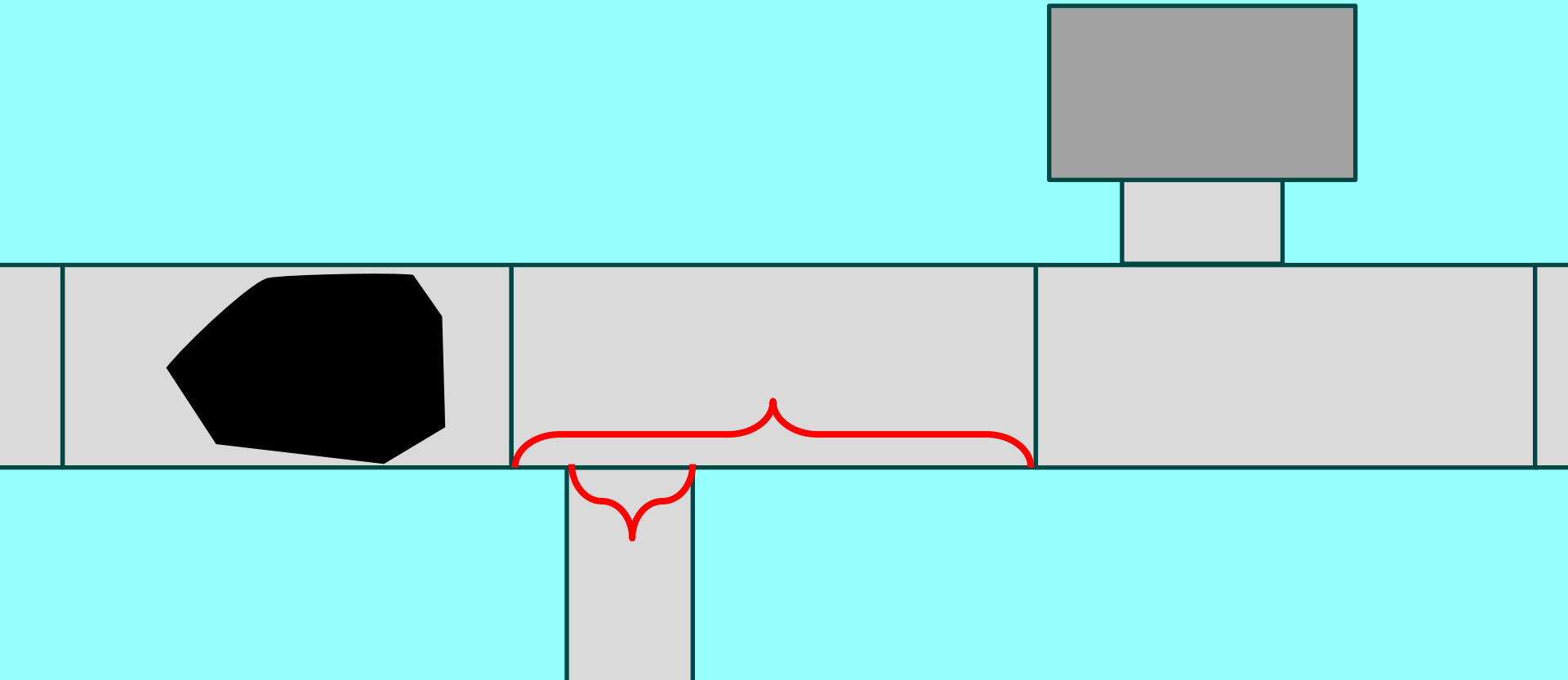
- Roadの隣接部分の線分短い方の中点と
  RoadとBuildingの中心を線で繋ぐ
- その線と瓦礫が接したら通行不可



POV（通行可能判断）

POVの条件

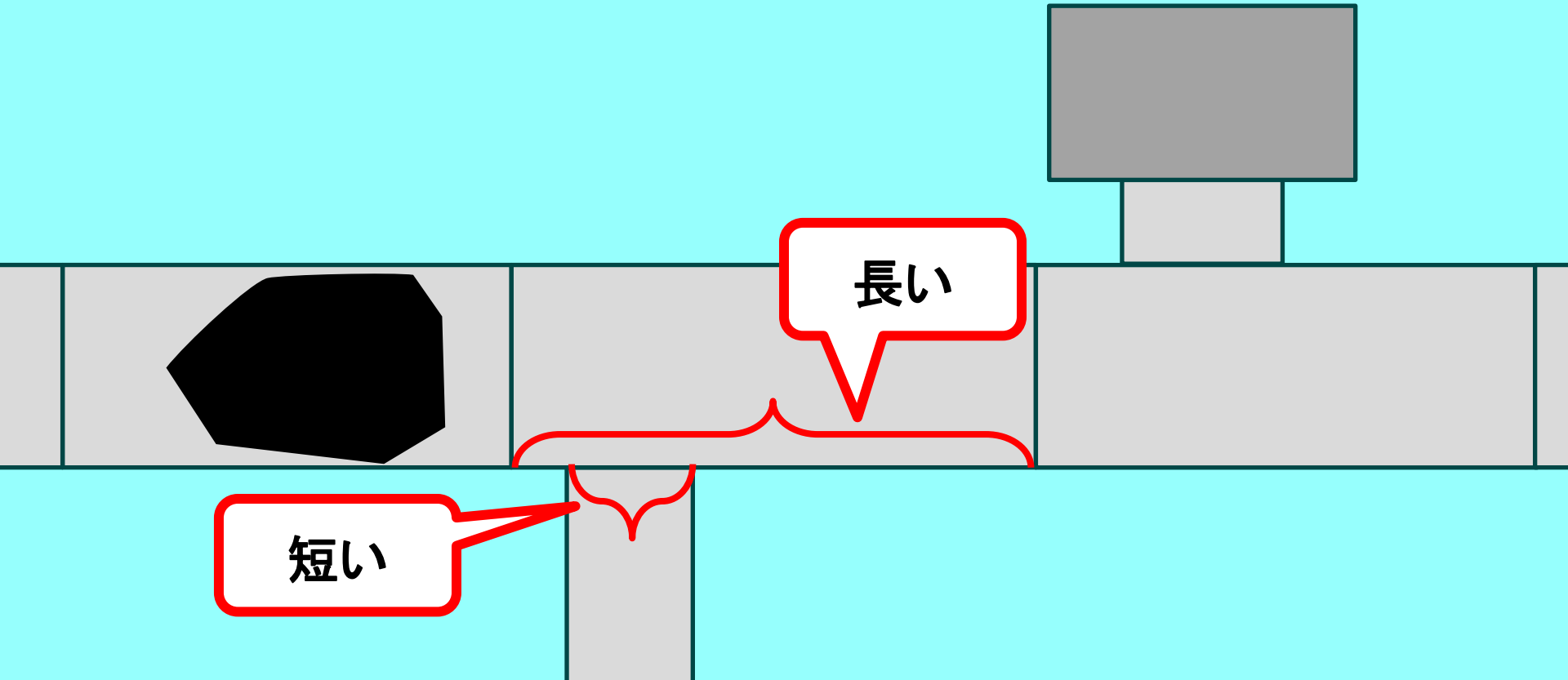
- Roadの隣接部分の線分短い方の中点と
  RoadとBuildingの中心を線で繋ぐ
- その線と瓦礫が接したら通行不可



POV（通行可能判断）

POVの条件

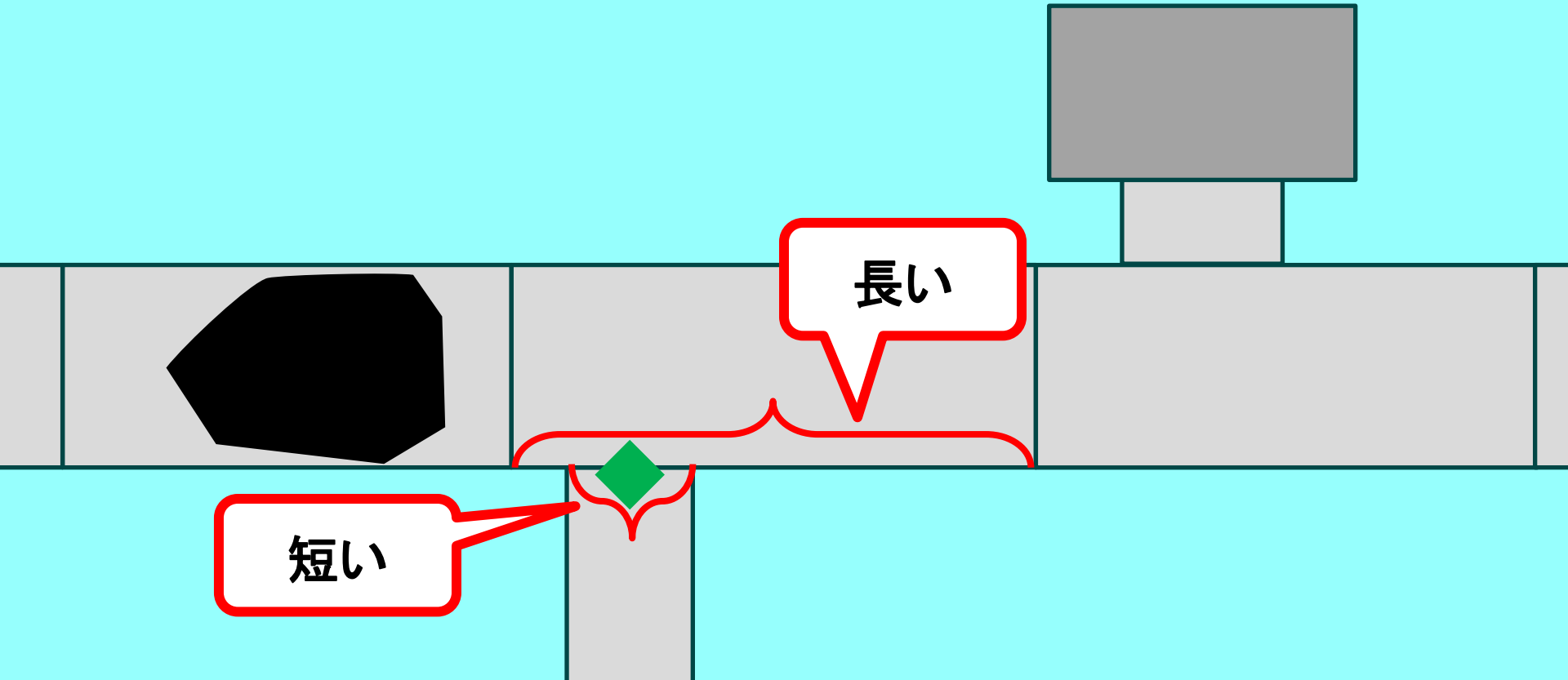
- Roadの隣接部分の線分短い方の中点とRoadとBuildingの中心を線で繋ぐ
- その線と瓦礫が接したら通行不可



POV（通行可能判断）

POVの条件

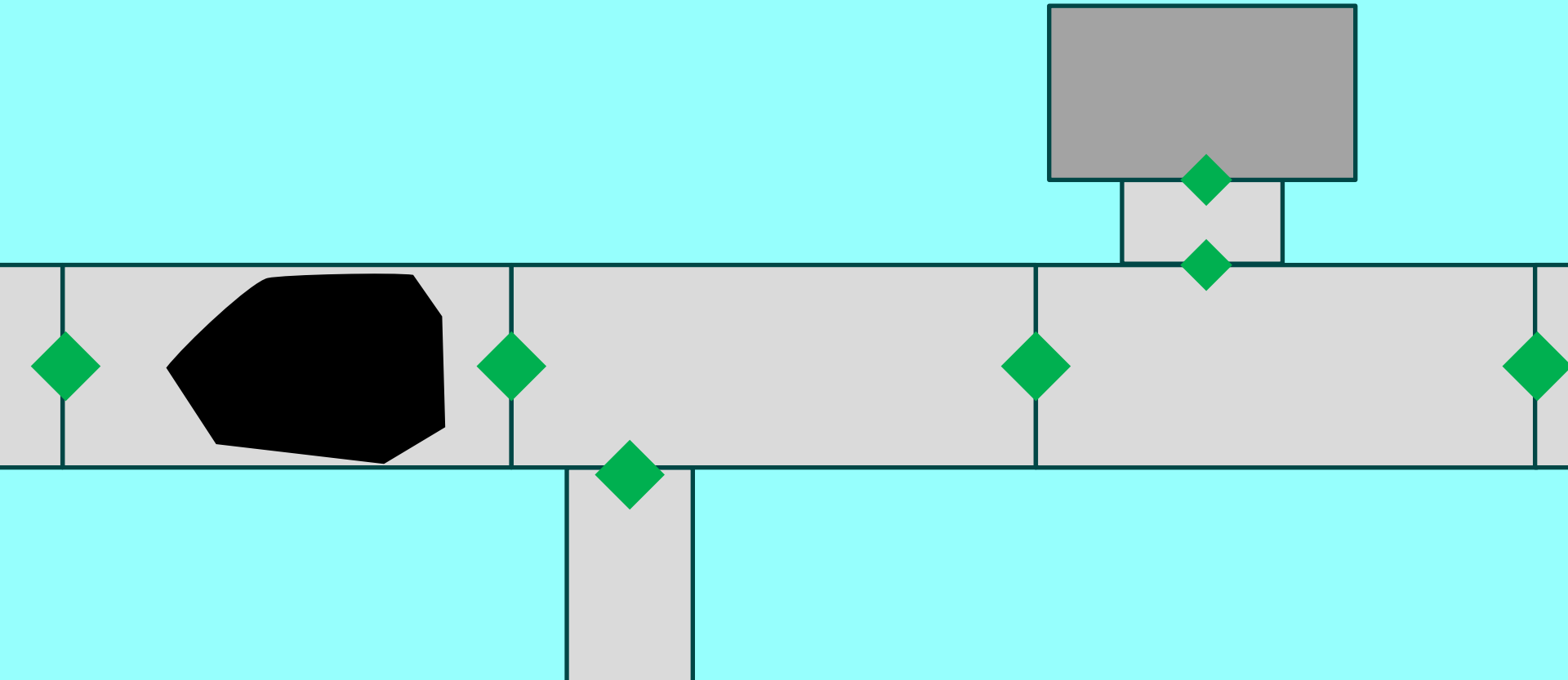
- Roadの隣接部分の線分短い方の中点と
  RoadとBuildingの中心を線で繋ぐ
- その線と瓦礫が接したら通行不可



POV（通行可能判断）

POVの条件

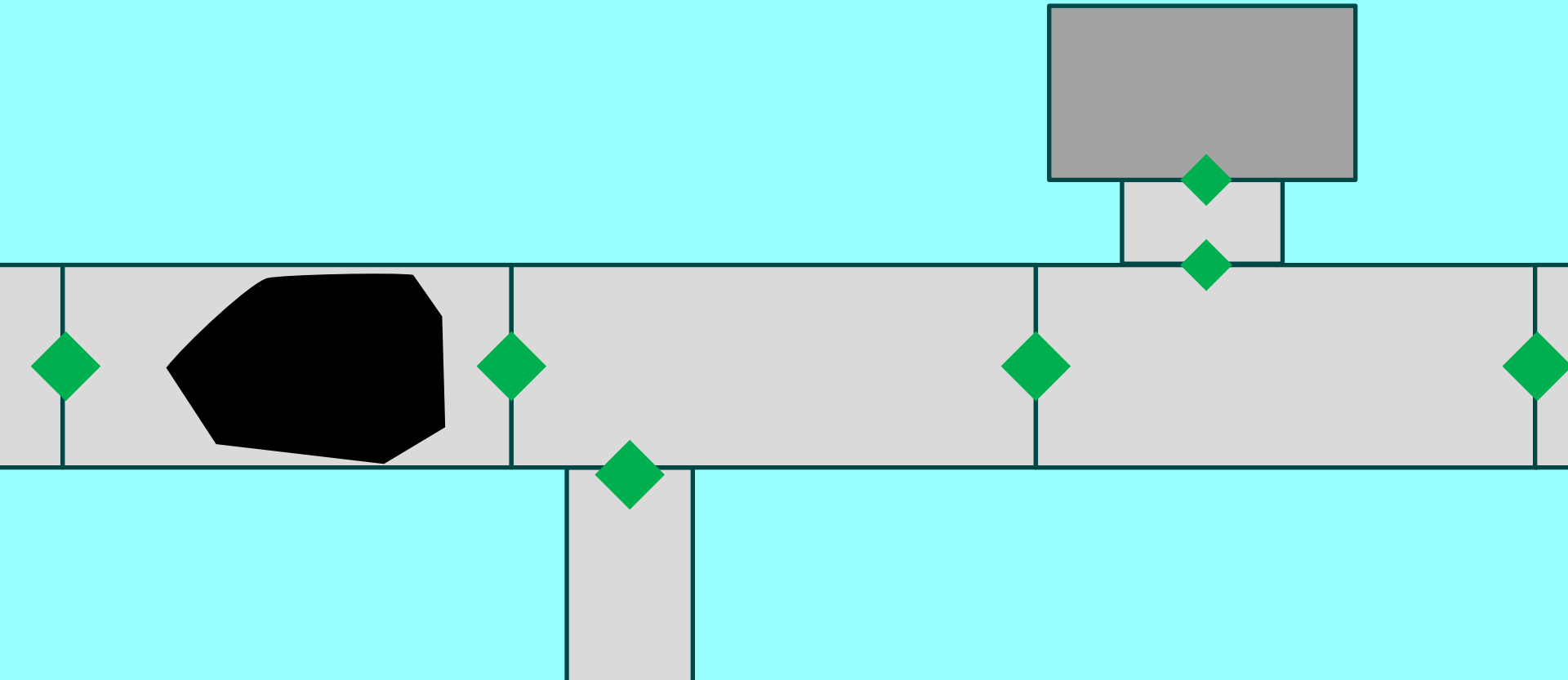
- Roadの隣接部分の線分短い方の中点と
  RoadとBuildingの中心を線で繋ぐ
- その線と瓦礫が接したら通行不可



POV（通行可能判断）

POVの条件

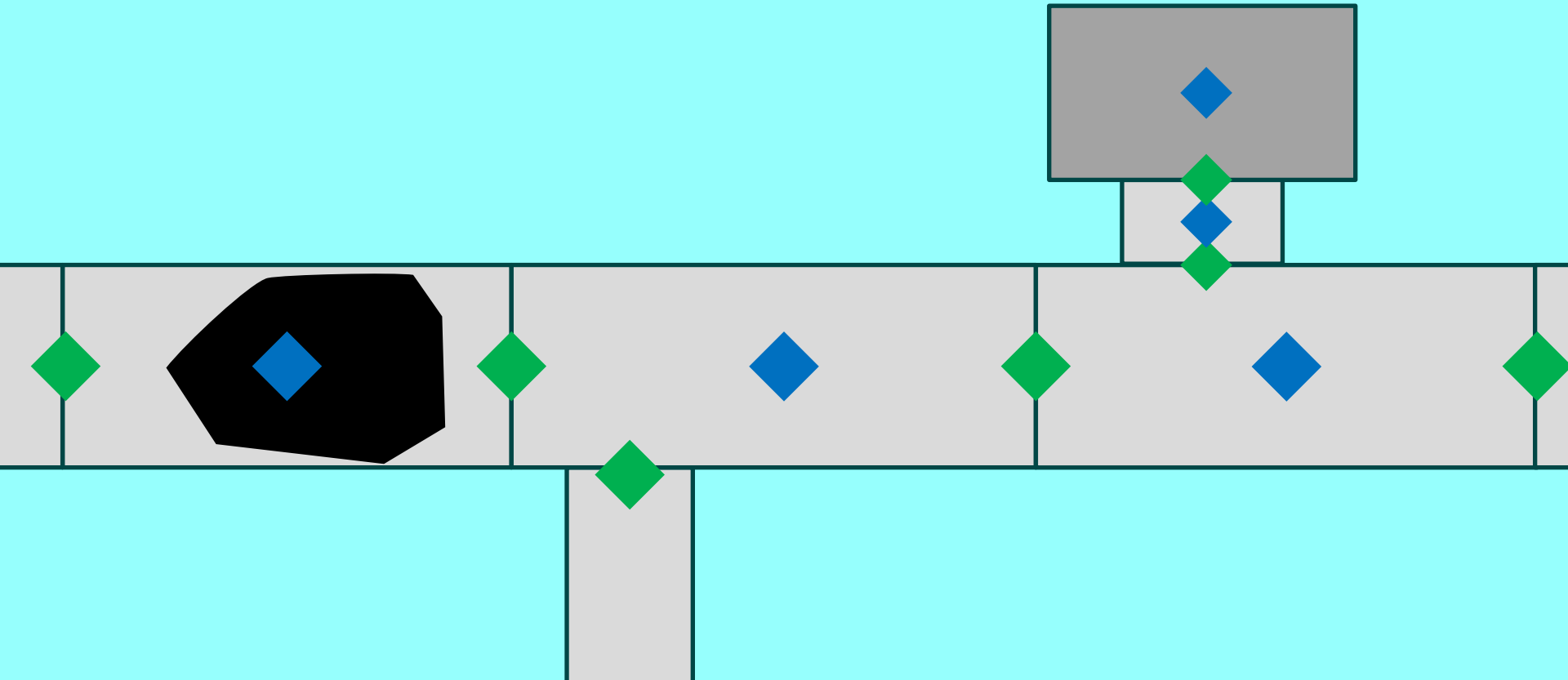
- Roadの隣接部分の線分短い方の中点と
  RoadとBuildingの中心を線で繋ぐ
- その線と瓦礫が接したら通行不可



POV（通行可能判断）

POVの条件

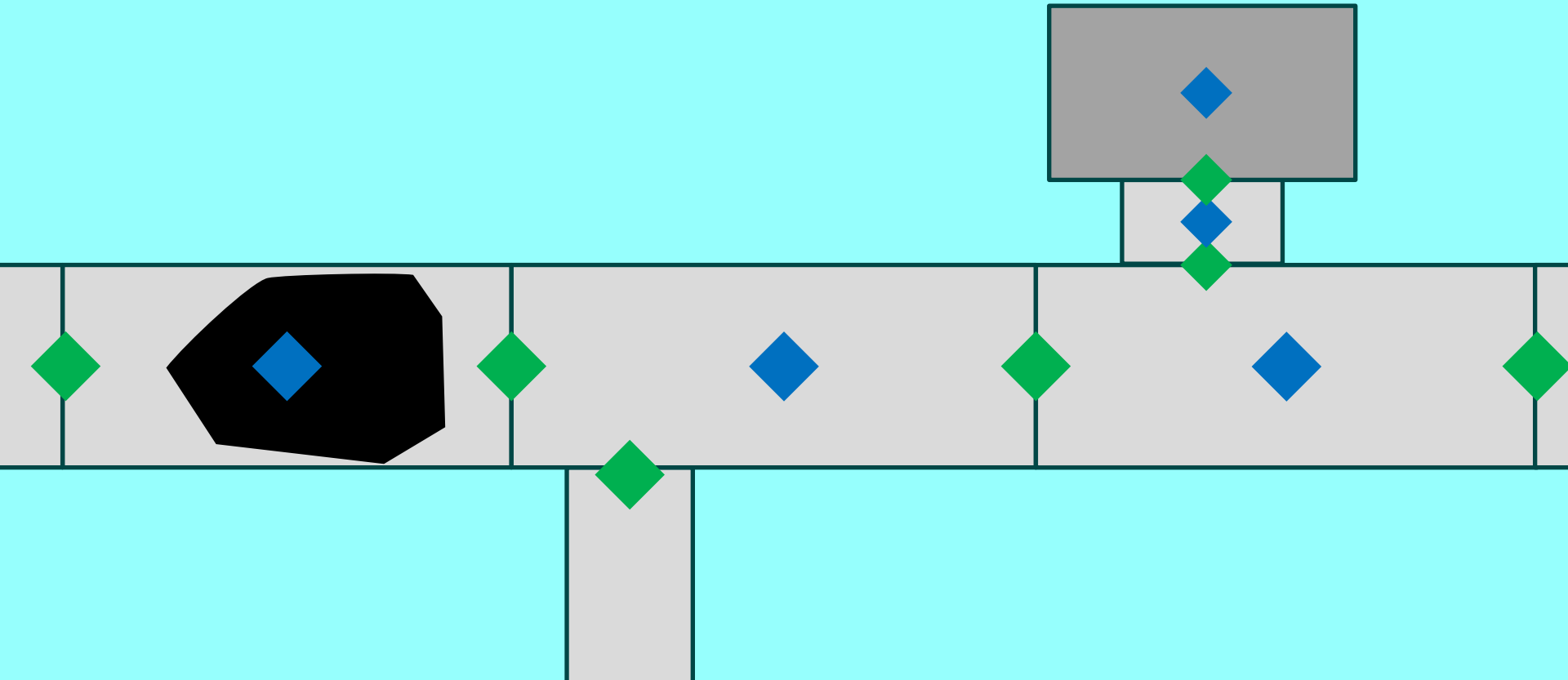
- Roadの隣接部分の線分短い方の中点と
  RoadとBuildingの中心を線で繋ぐ
- その線と瓦礫が接したら通行不可



POV（通行可能判断）

POVの条件

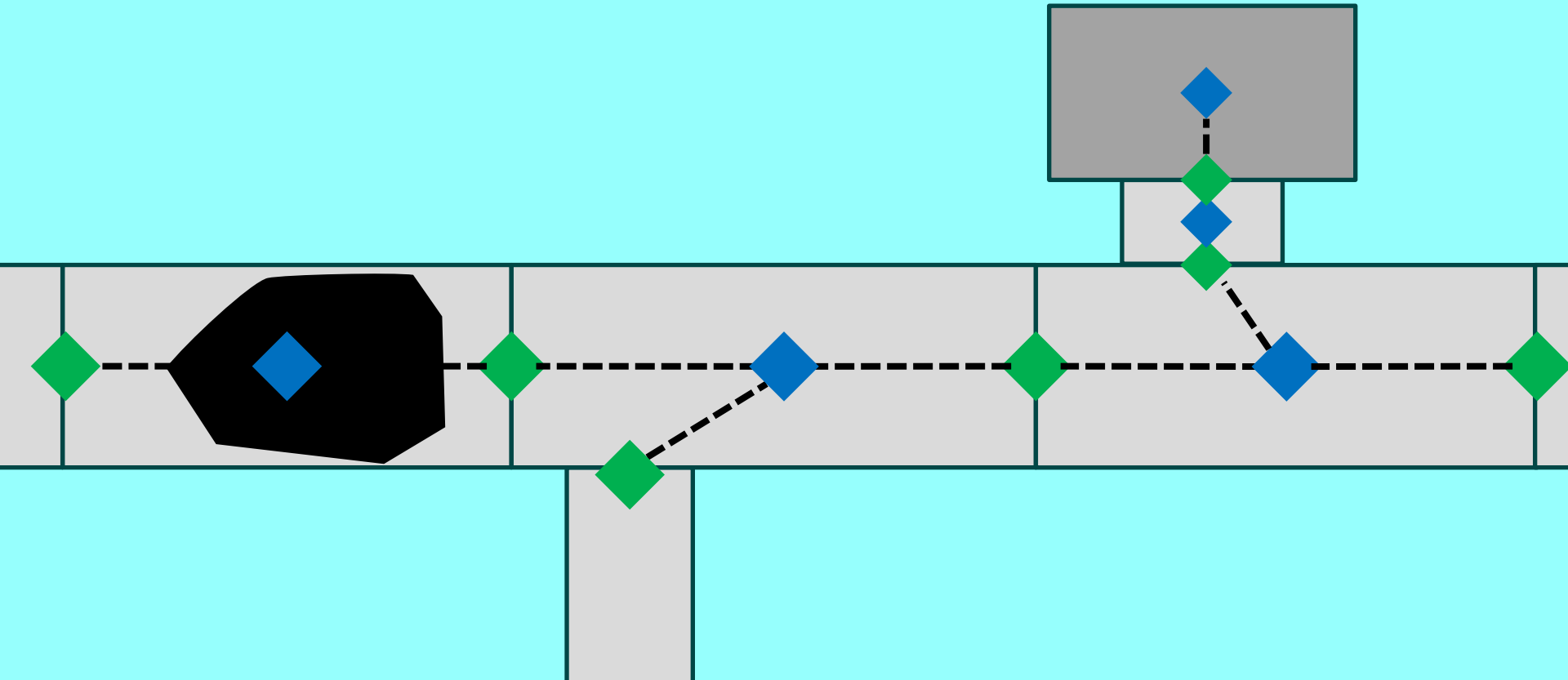
・Roadの隣接部分の線分短い方の中点と
RoadとBuildingの中心を**線で繋ぐ**
・その線と瓦礫が接したら通行不可



POV（通行可能判断）

POVの条件

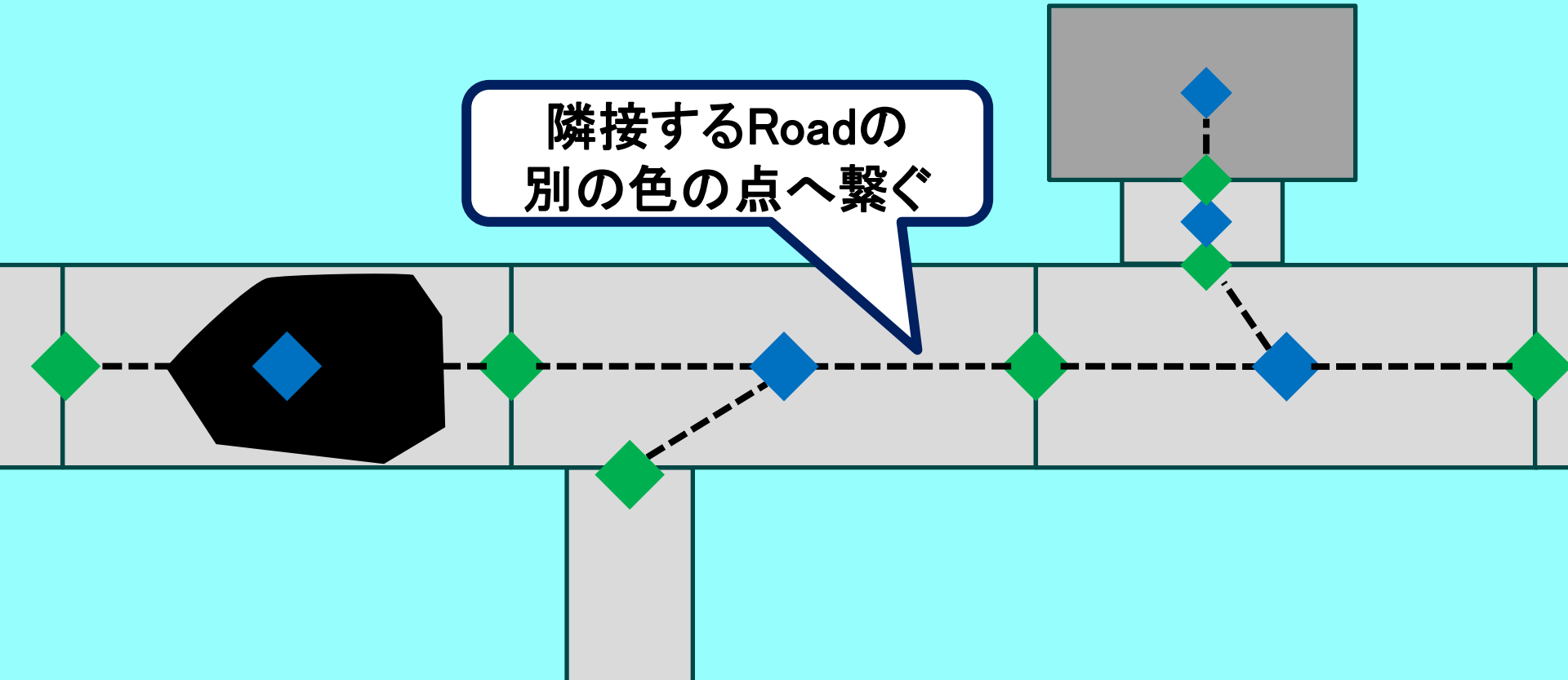
・Roadの隣接部分の線分短い方の中点と
RoadとBuildingの中心を**線で繋ぐ**
・その線と瓦礫が接したら通行不可



POV（通行可能判断）

POVの条件

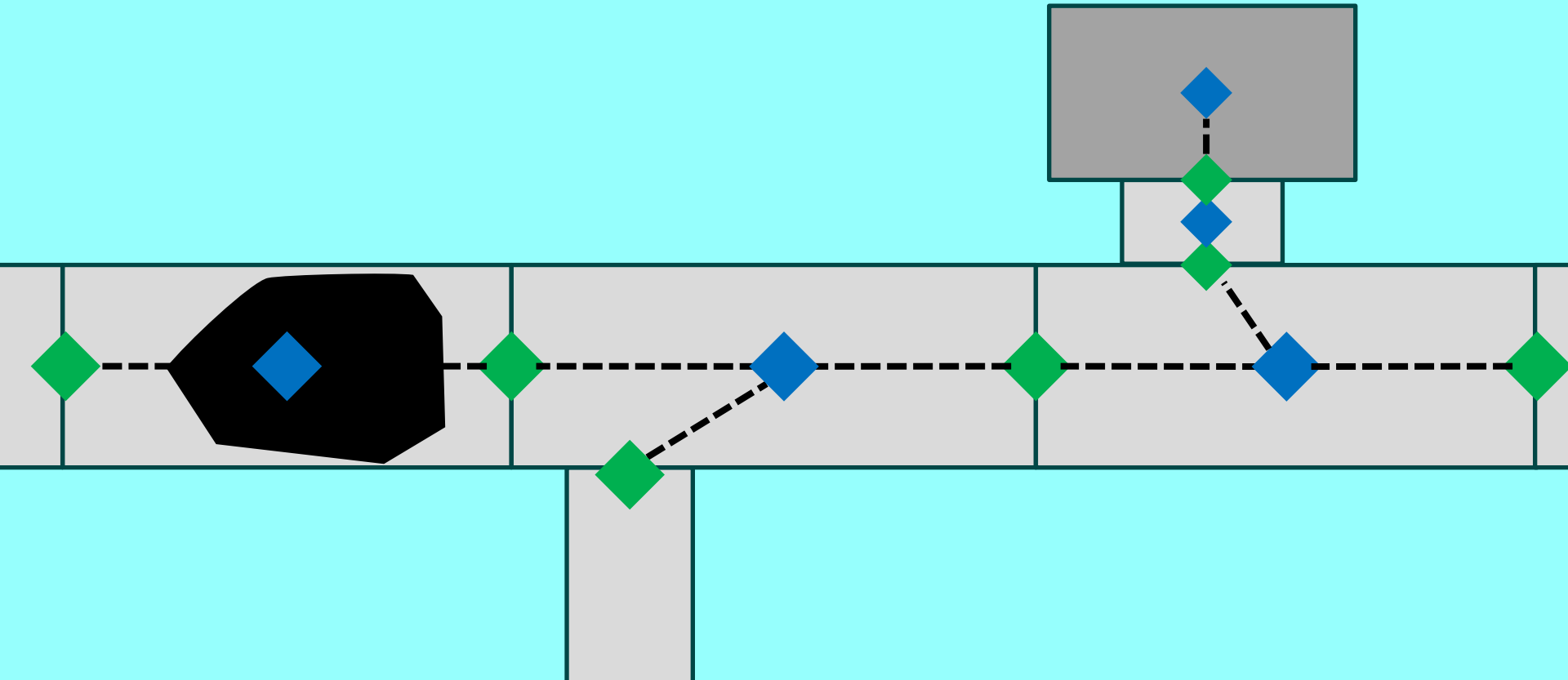
・Roadの隣接部分の線分短い方の中点と
RoadとBuildingの中心を**線で繋ぐ**
・その線と瓦礫が接したら通行不可



POV（通行可能判断）

POVの条件

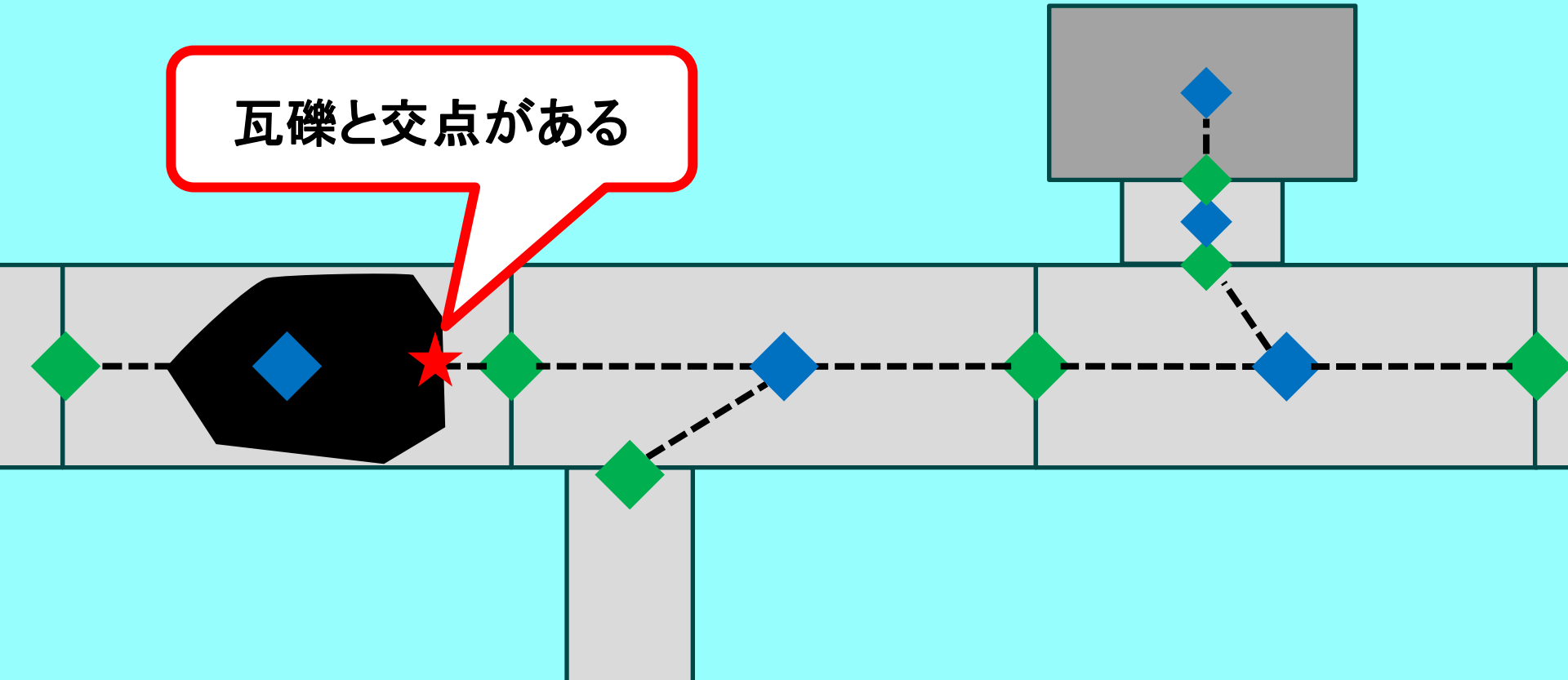
- Roadの隣接部分の線分短い方の中点とRoadとBuildingの中心を線で繋ぐ
- その線と瓦礫が接したら通行不可



POV（通行可能判断）

POVの条件

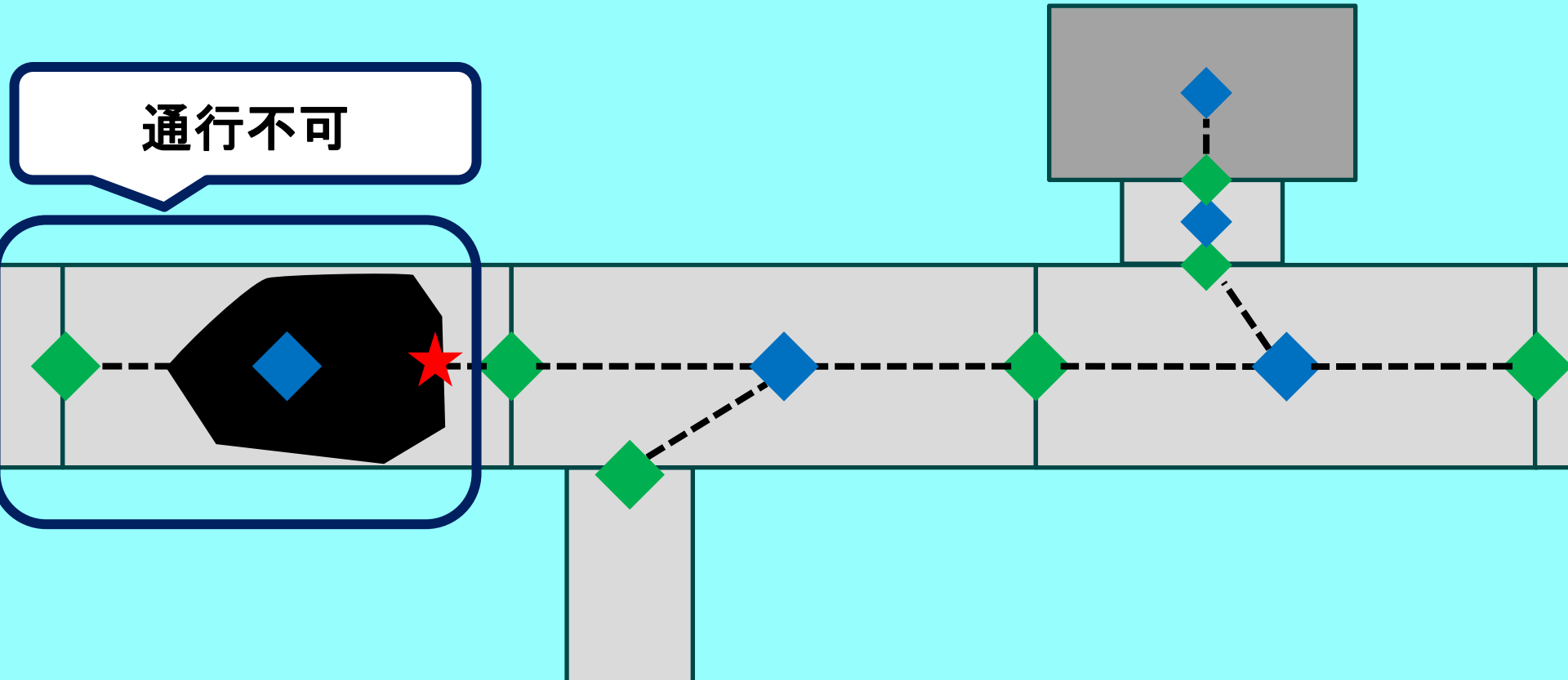
- Roadの隣接部分の線分短い方の中点とRoadとBuildingの中心を線で繋ぐ
- その線と瓦礫が接したら通行不可



POV（通行可能判断）

POVの条件

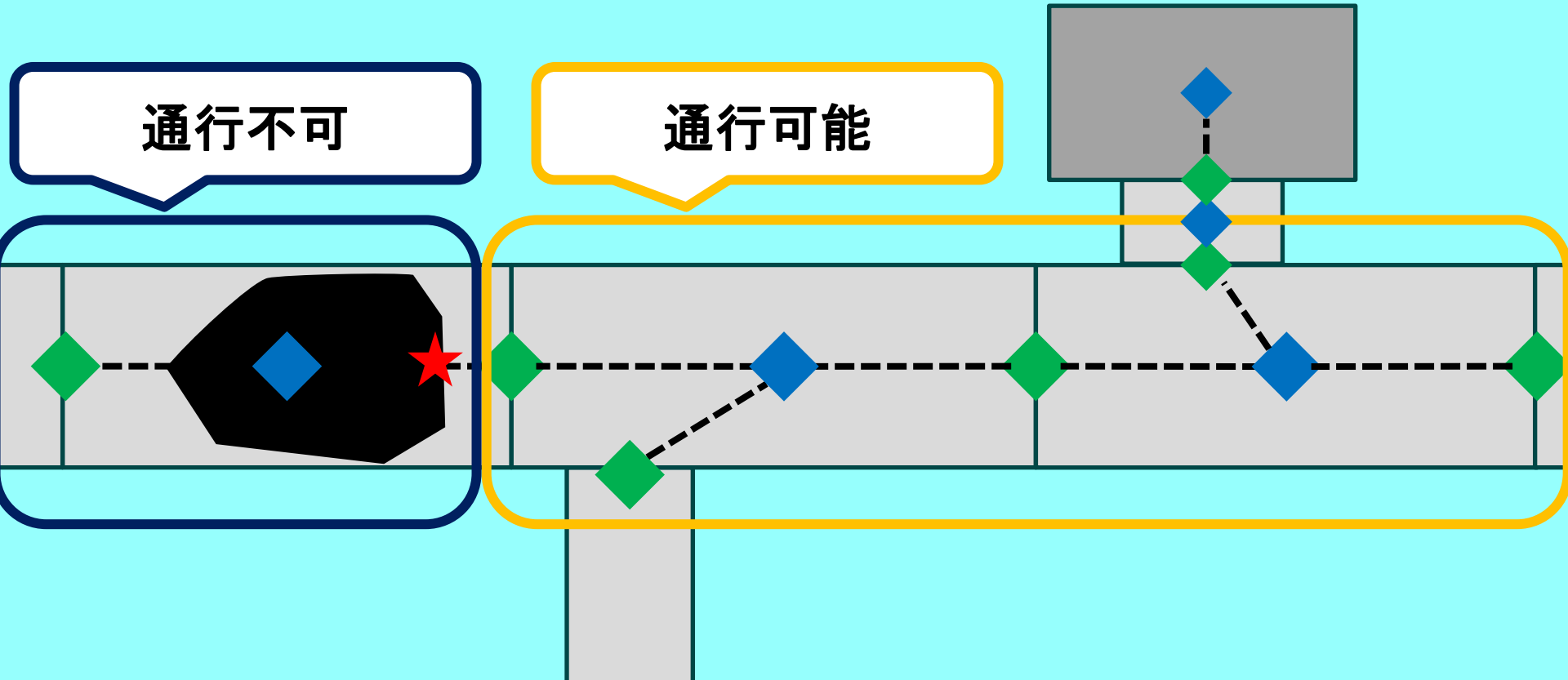
- Roadの隣接部分の線分短い方の中点とRoadとBuildingの中心を線で繋ぐ
- その線と瓦礫が接したら通行不可

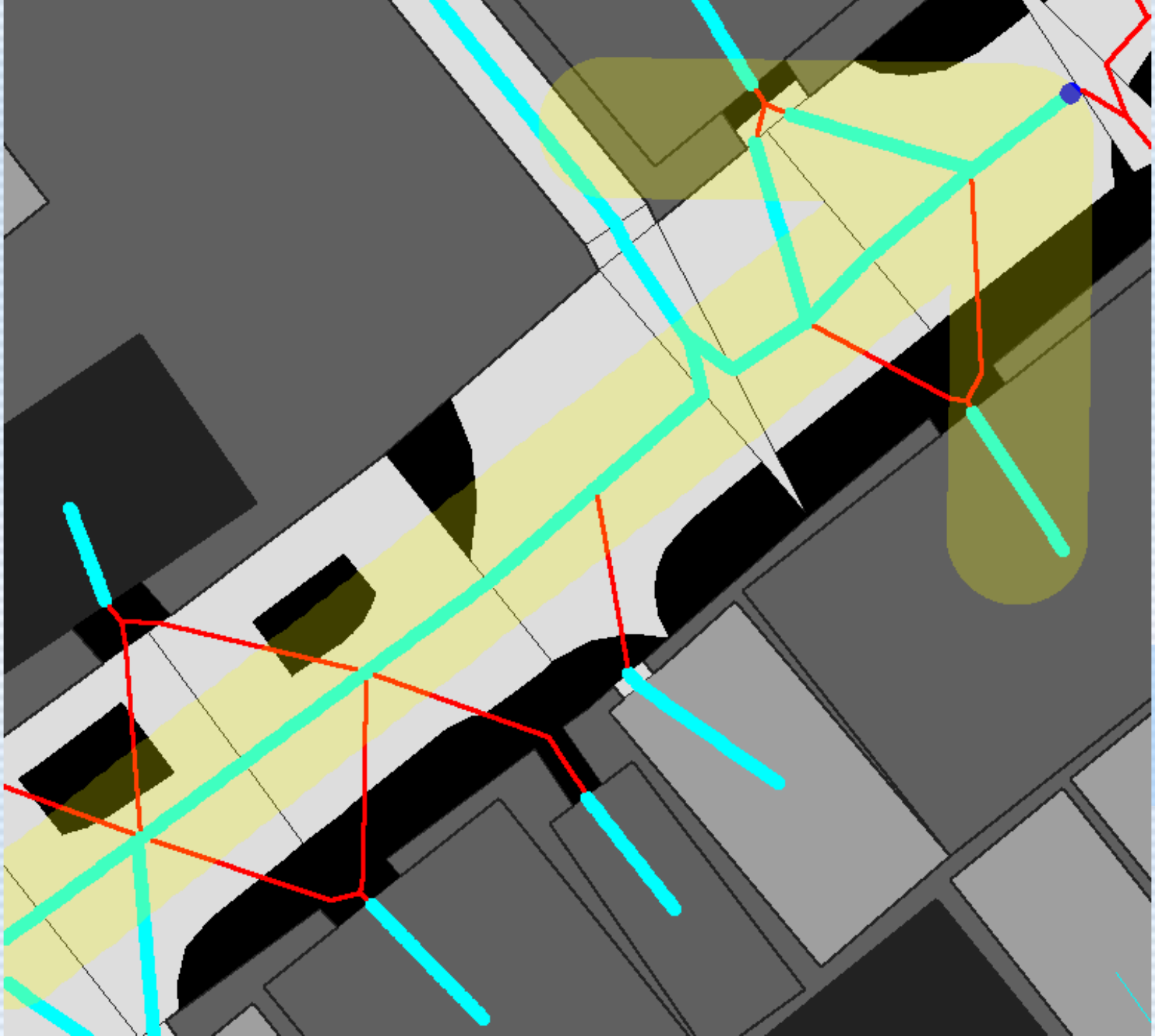


POV（通行可能判断）

POVの条件

- Roadの隣接部分の線分短い方の中点とRoadとBuildingの中心を線で繋ぐ
- その線と瓦礫が接したら通行不可





POVによるPFの進化

今まで

周りにある瓦礫はすべて除去

今

通行可能なら除去しなくて良い

POVによるPFの進化

今まで

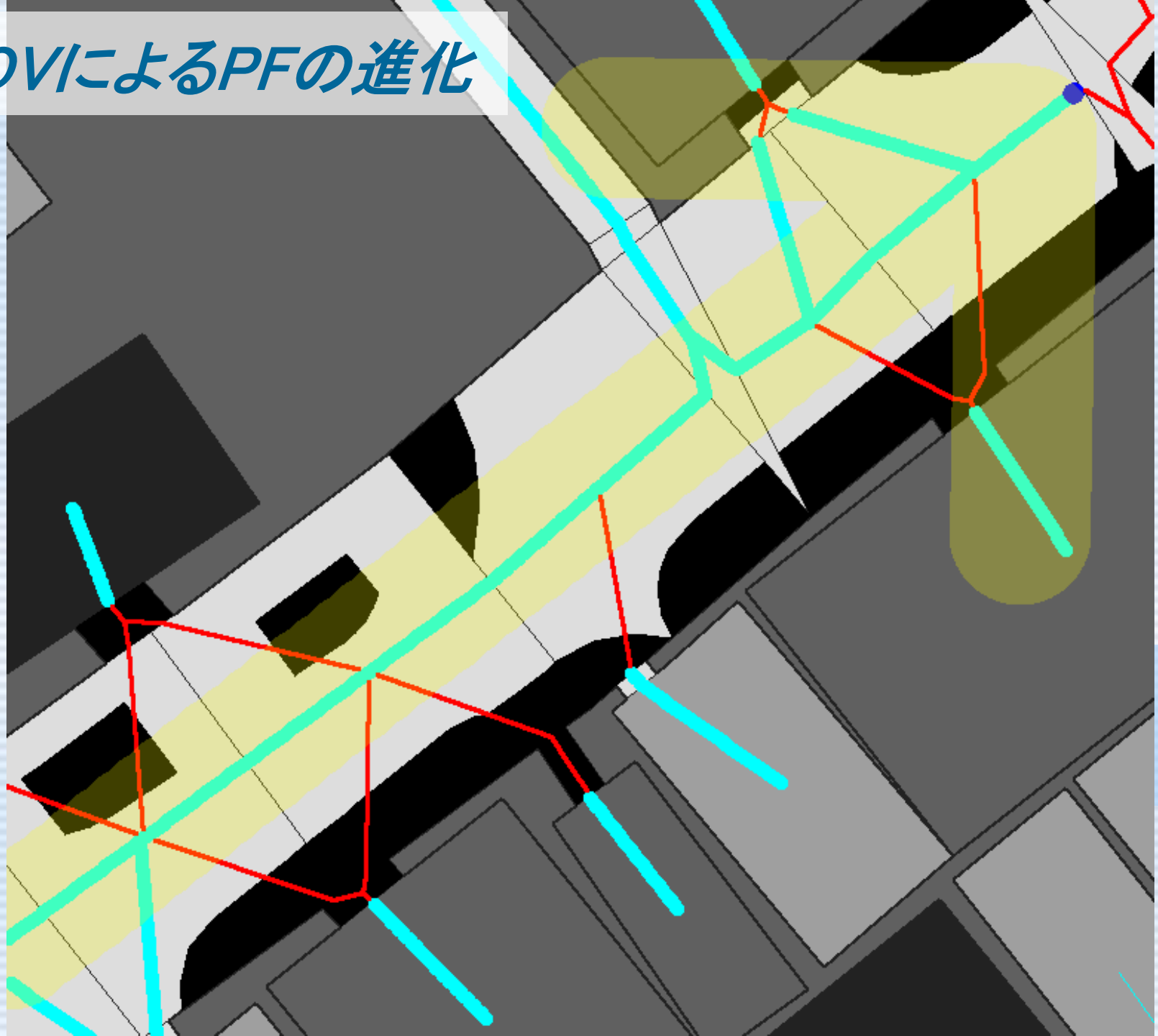
周りにある瓦礫はすべて除去

無駄な動きを
減らす

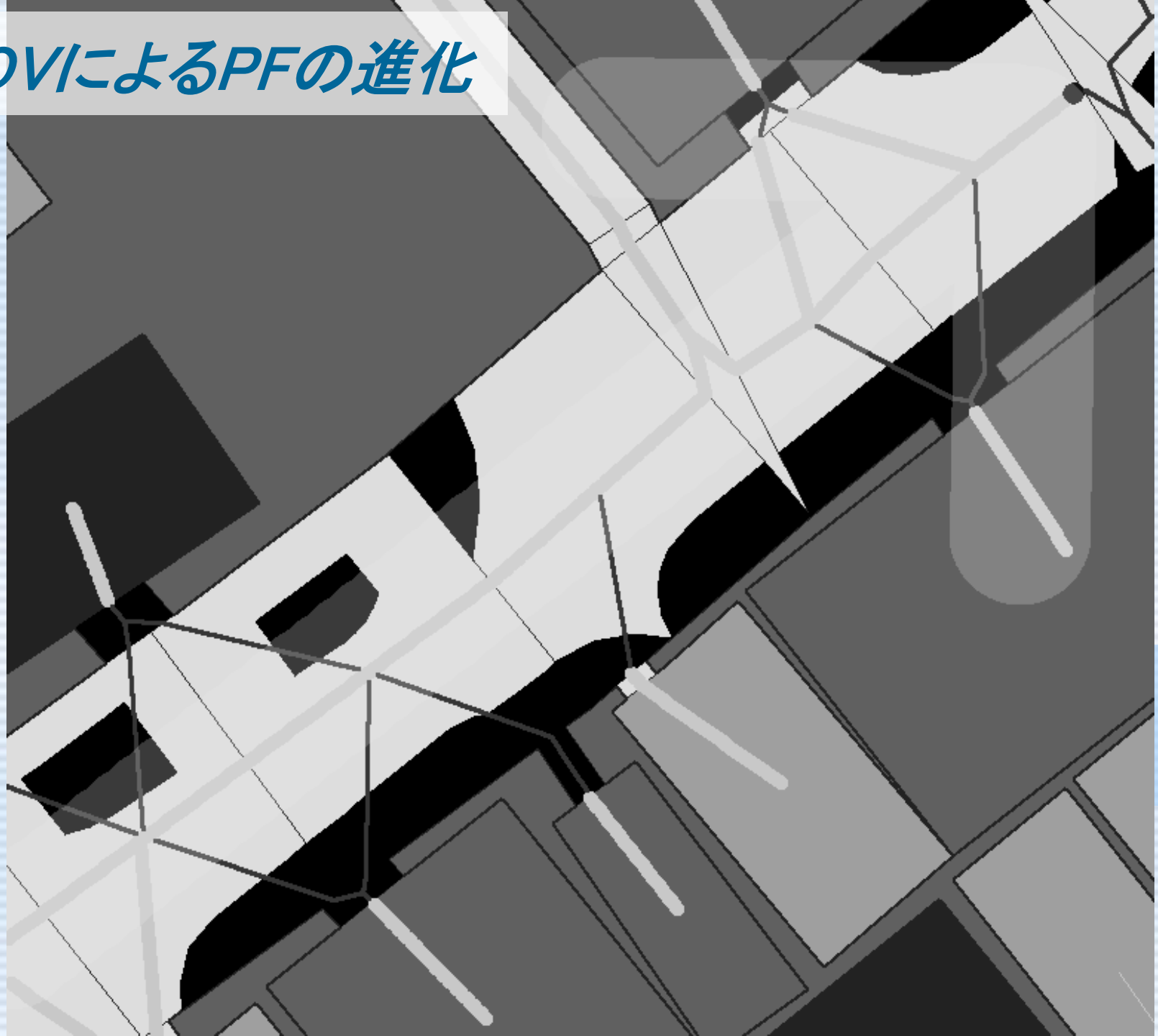
今

通行可能なら除去しなくて良い

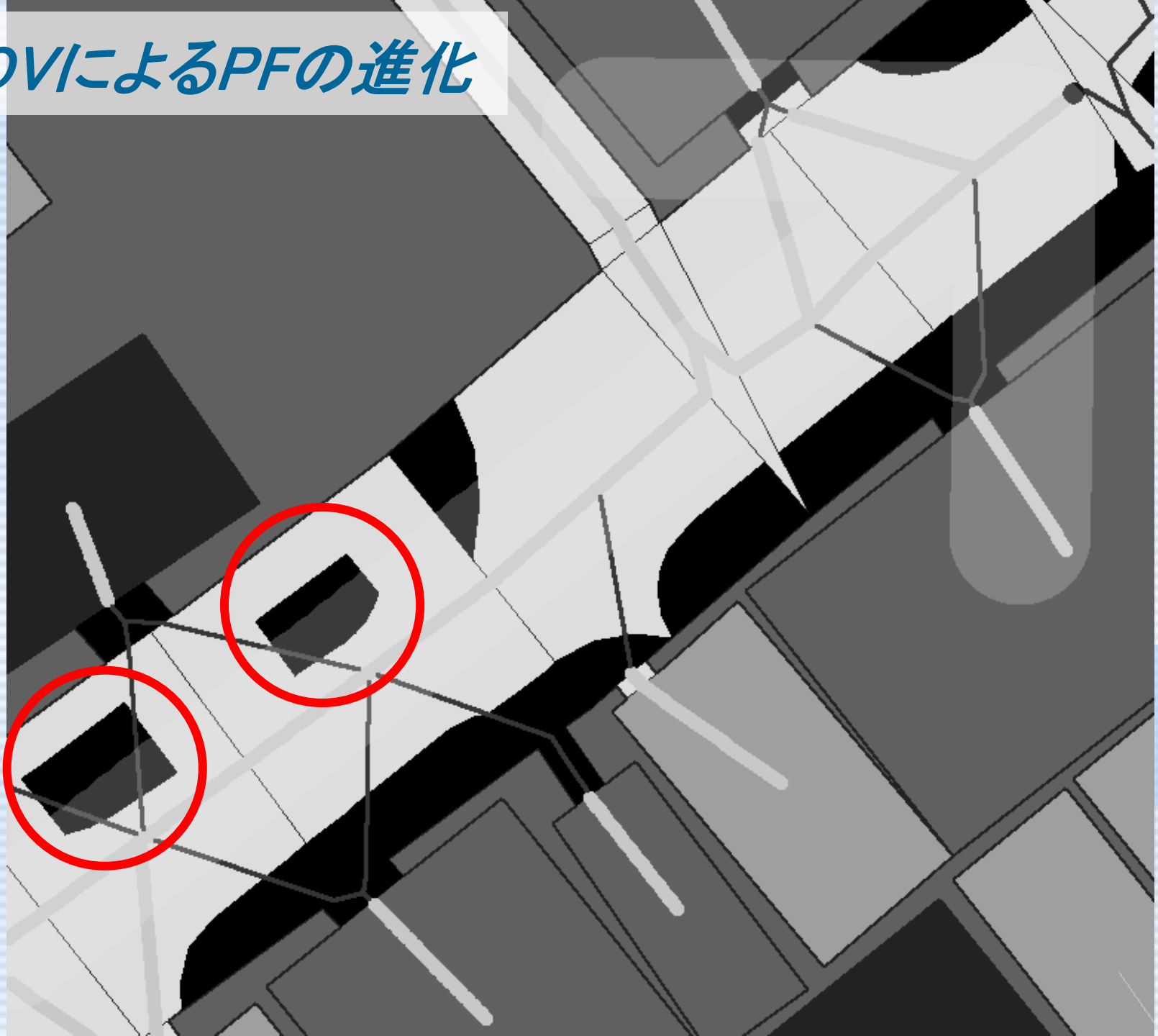
POVによるPFの進化



POVによるPFの進化



POVによるPFの進化



POVによるPFの進化

POVによるPFの進化

結果

- 的確な経路探索
- PFの瓦礫除去を必要最低限に抑える

課題

- 通行可能のパスも通行不可と判断する場合がある

ご清聴ありがとうございました

使用画像

eclipse

入手先<http://www.eclipse.org/>(アクセス日 : 2012/11/18).

Subversion

入手先<http://subversion.apache.org/>(アクセス日 : 2012/11/18).

使用デザイン

sky.pot

入手先<http://gift.her.jp/pp_template/>(アクセス日 : 2012/11/10).