通过修正的渗透深度和引导检索解决二维排样问题

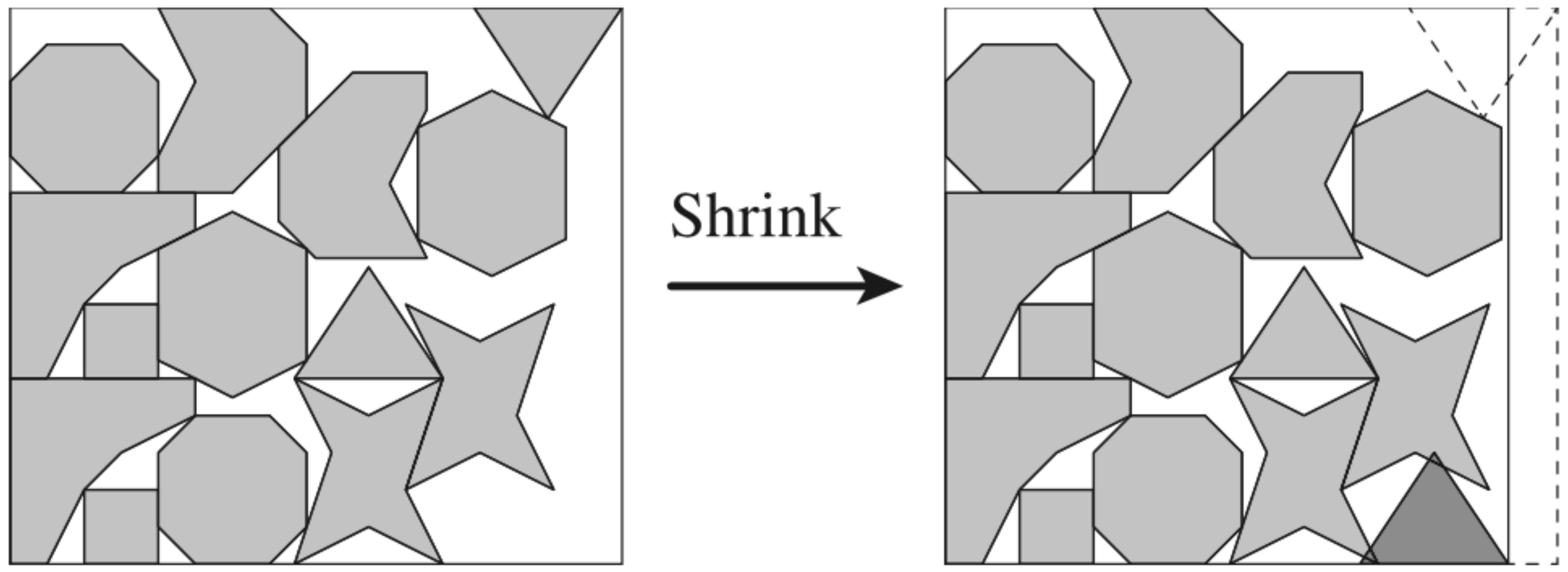
Use modified penetration depth and guided search to solve nesting problem

羊山, 王子路

2020.6.28

研究简介

Introduction to our research



基于布局的优化

在二维排样问题中，逐步并重复为每个形状寻找产生重叠更小的位置，从而实现整体优化，这是现在的最优思路

二维排样问题 [2D irregular packing problem]

一种常见的组合优化问题，一般通过左底部算法获得初始解，再通过基于布局优化寻找更优的解（如上），该问题为NP-Hard。

研究问题

在为形状寻找更优位置时，现阶段算法均只能够实现局部检索或近似全局检索，导致计算效率低下且效果不佳

我们做了什么?

1. 建立了全局线性检索模型

首先证明了凸多边形全局检索可行性，其次通过对重叠评价标准进行调整，建立了凹多边形的全局检索模型。

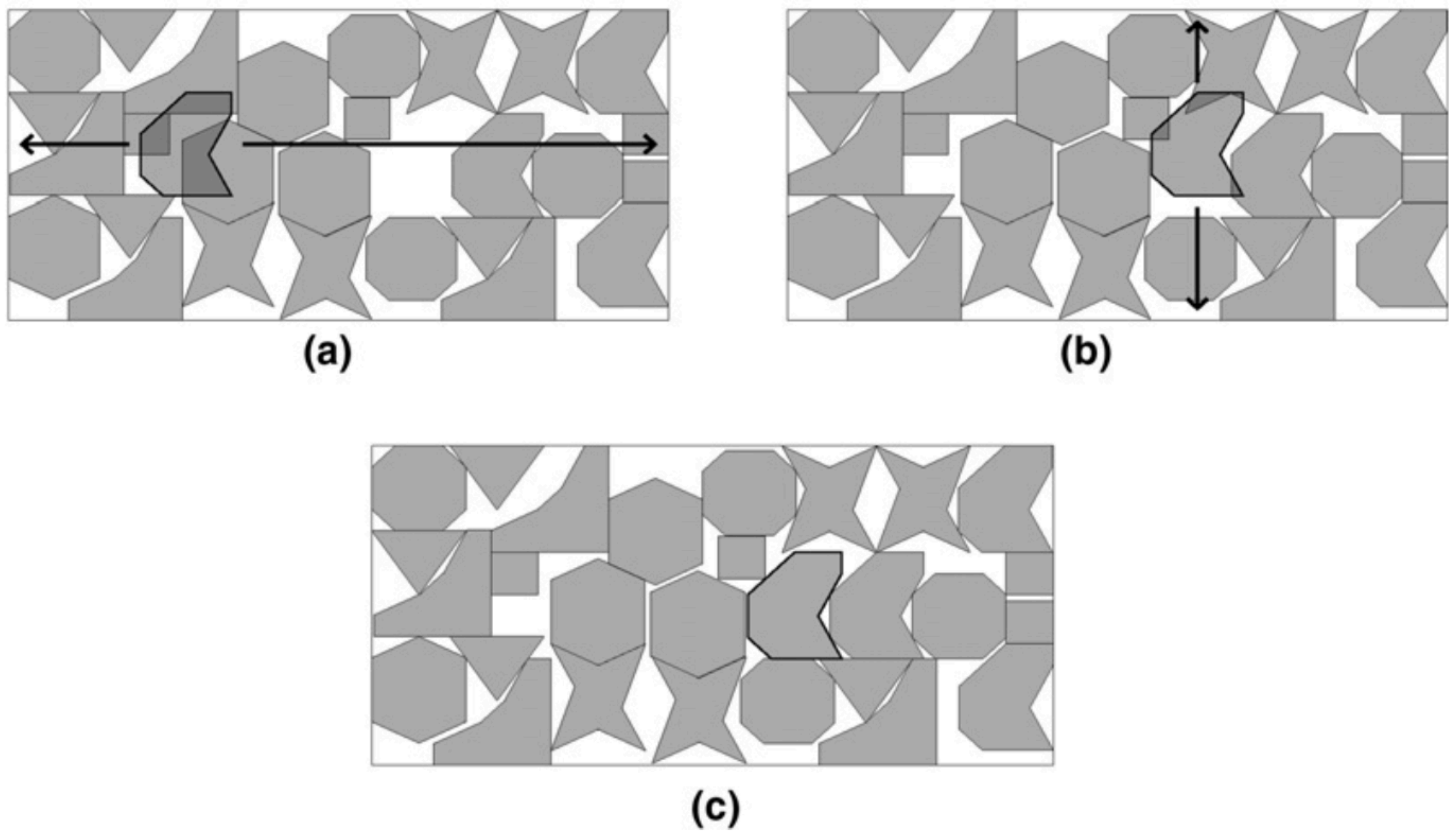
2. 在凸多边形实现了接近历史最优利用率

在凸多边形数据集fu上实现了90%±0.5%的实验效果，接近近年算法最佳结果91%-92%[7][8][9][10][11]，同时需要检索位置的数目下降了80%[7]

理论上可以实现而非接近最佳结果，原因正在分析中

相关研究

Related researches



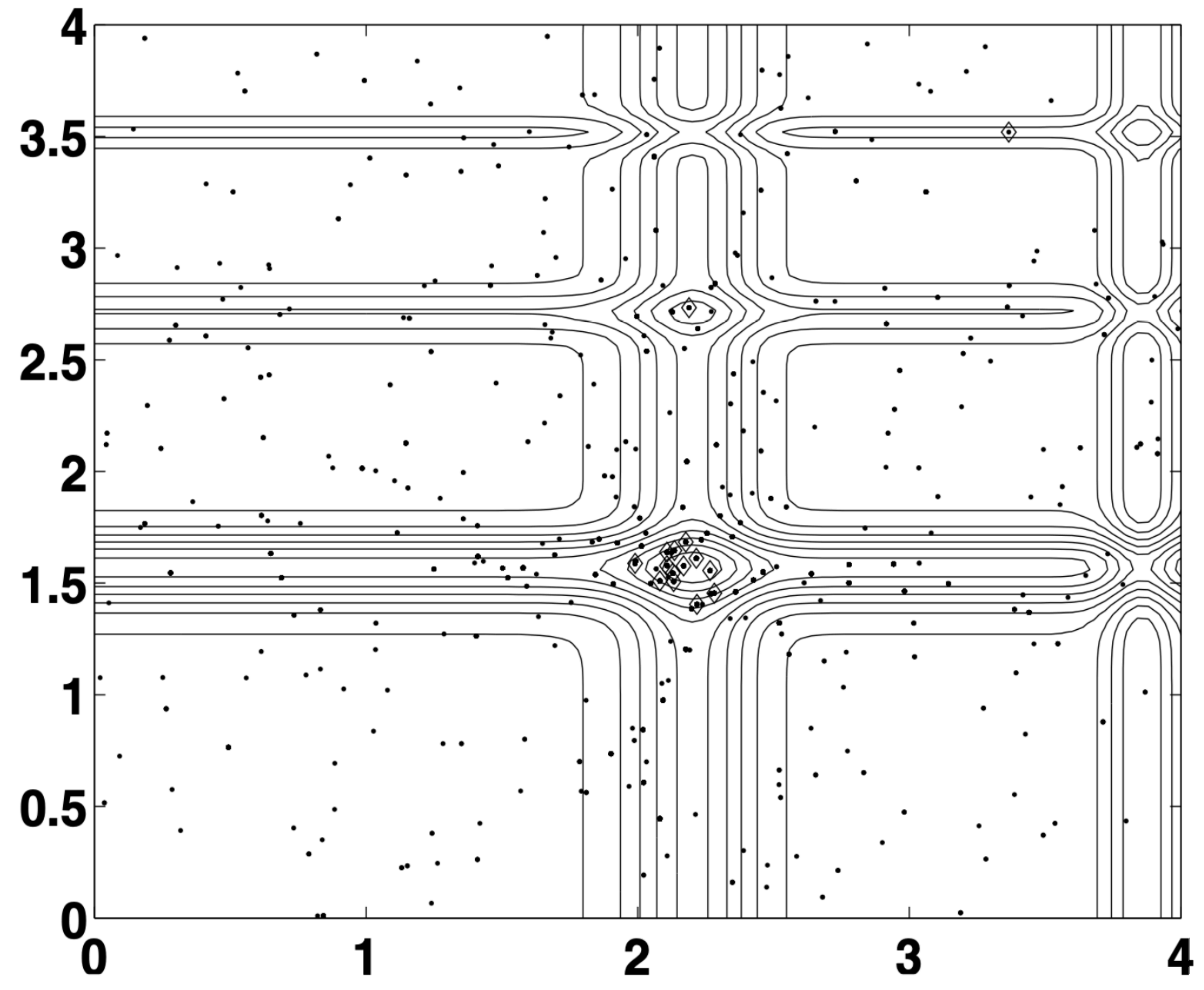


Fig. 3. Search paths of nests using Cuckoo Search. The final locations of the nests are marked with ◇ in the figure.

引导邻域检索

随机选择形状，通过水平和垂直平移到重叠最小的地方以实现优化，同时在计算总重叠的时候，调整不同形状的重叠权重，避免陷入局部最优[11]

改进的邻域检索

上图中引导布谷鸟检索对邻域检索进行了改进，通过布谷鸟检索算法，在全局寻找最优位置，每次检索450个点[7]，所有改进方式均只对局部检索范围进行了延伸[8][9][10]

几何概念

Geometry concept

TONGJI UNIVERSITY

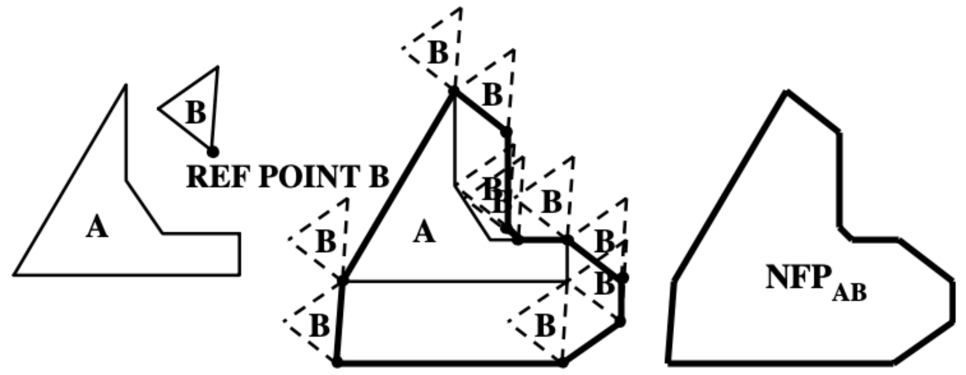


Fig. 1. The no-fit polygon of two shapes A and B.

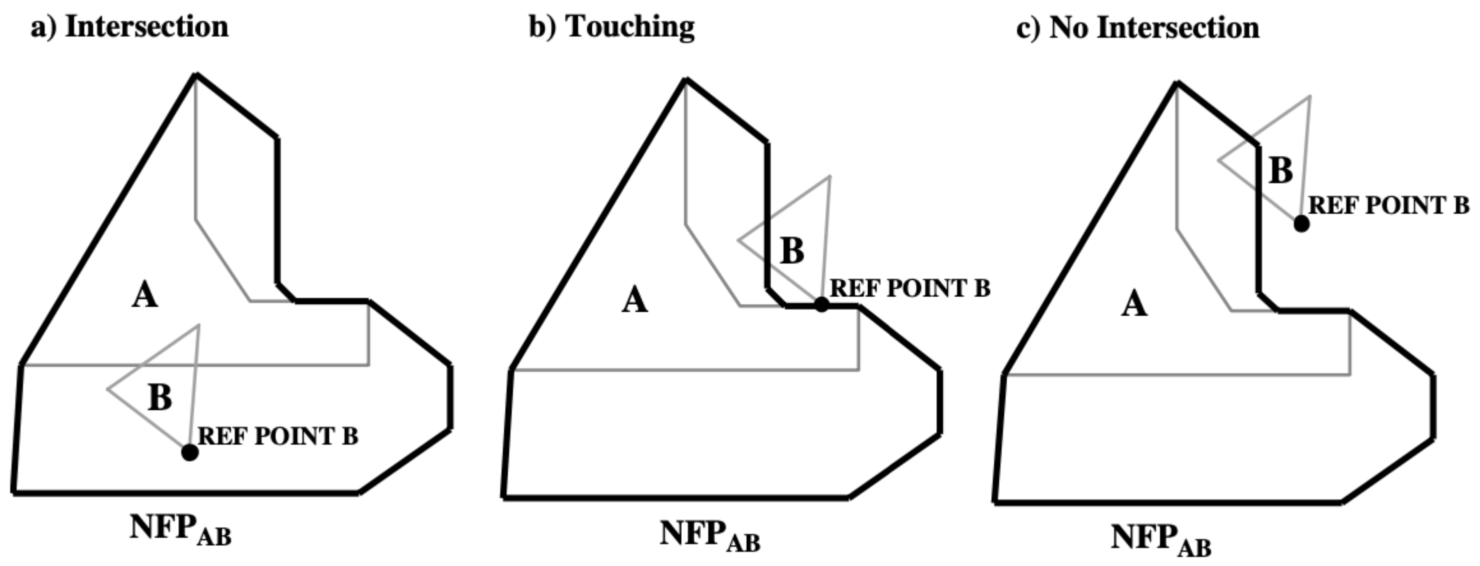


Fig. 2. Using the no-fit polygon to test for intersection between polygons A and B.

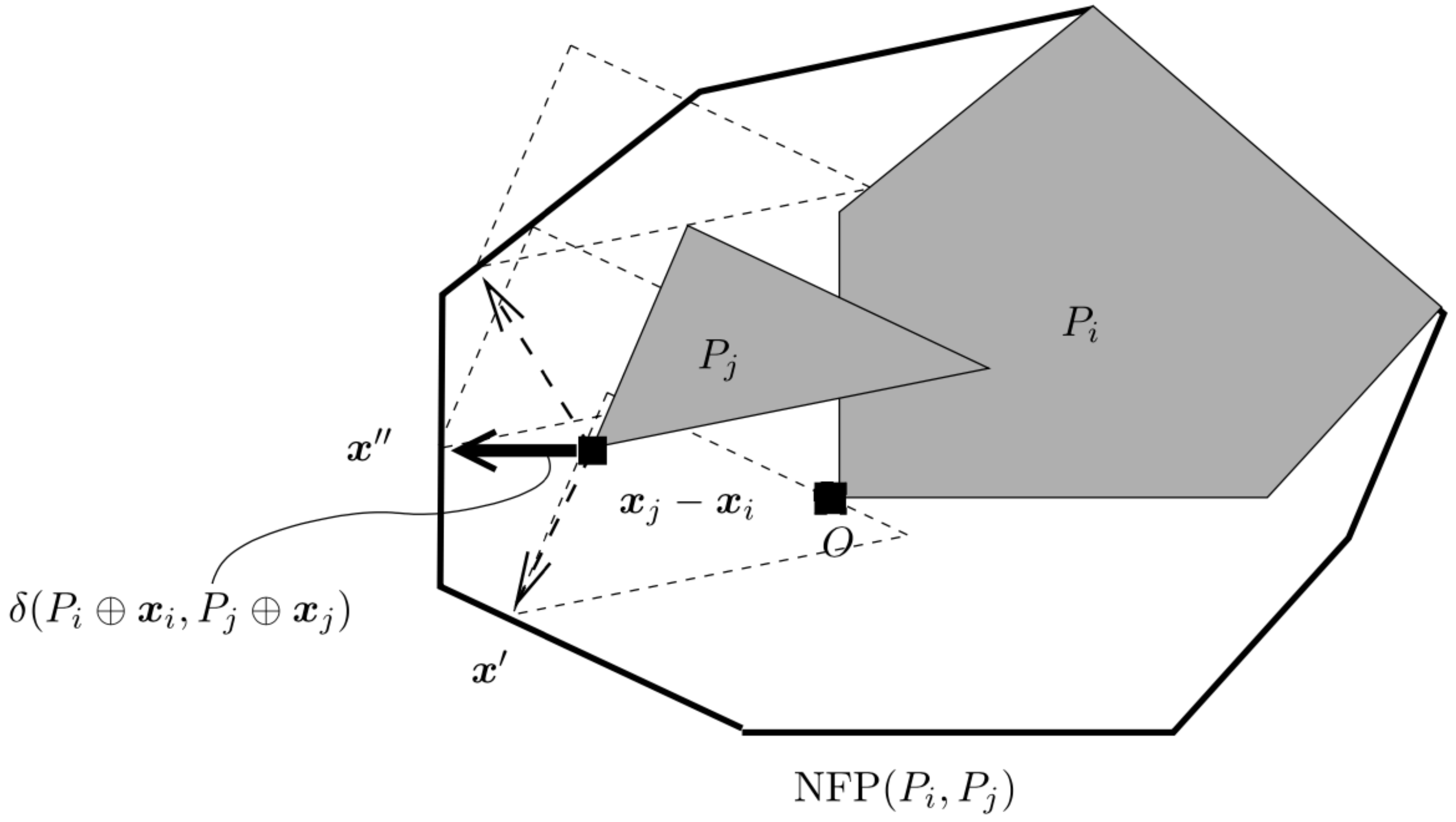


Fig. 6. The no-fit polygon NFP(P_i, P_j) and the penetration depth $\delta(P_i \oplus \boldsymbol{x}_i, P_j \oplus \boldsymbol{x}_j)$.

NFP[No-fit Polygon]

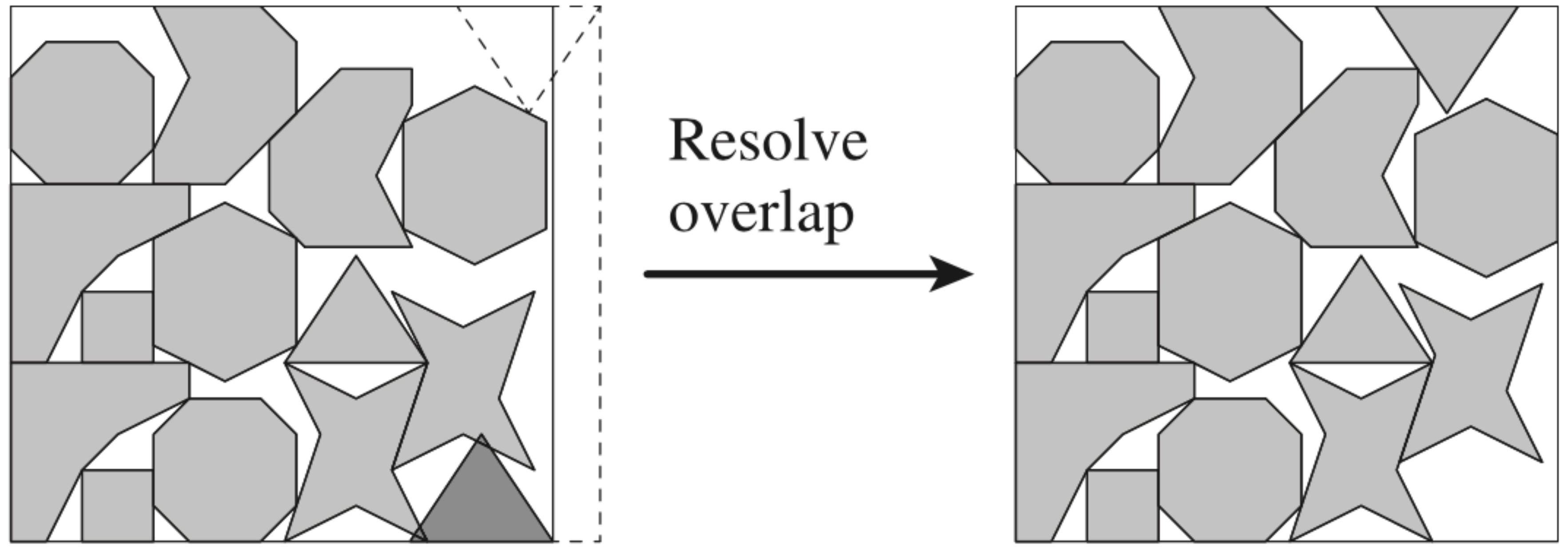
NFP用于判断形状是否重叠。上图所示，NFP$_{AB}$是形状B绕形状A一圈，其参考点REF POINT B的轨迹，若该点在NFP$_{AB}$内，则形状重叠[1][4]

PD[Penetration Depth]

渗透深度一般用于替代重合，计算复杂度为θ(Number of NFP's Edges)。上图中，PD即P_j与P_i没有重叠P_j所需要移动的最短距离，为P_j参考点到NFP所有边的最短距离[7][8][10]

如何检索潜在的重叠更小的位置

How to search for the potential best position



问题：如何为深色的三角形寻找到重叠更小的位置?

检索过程本质

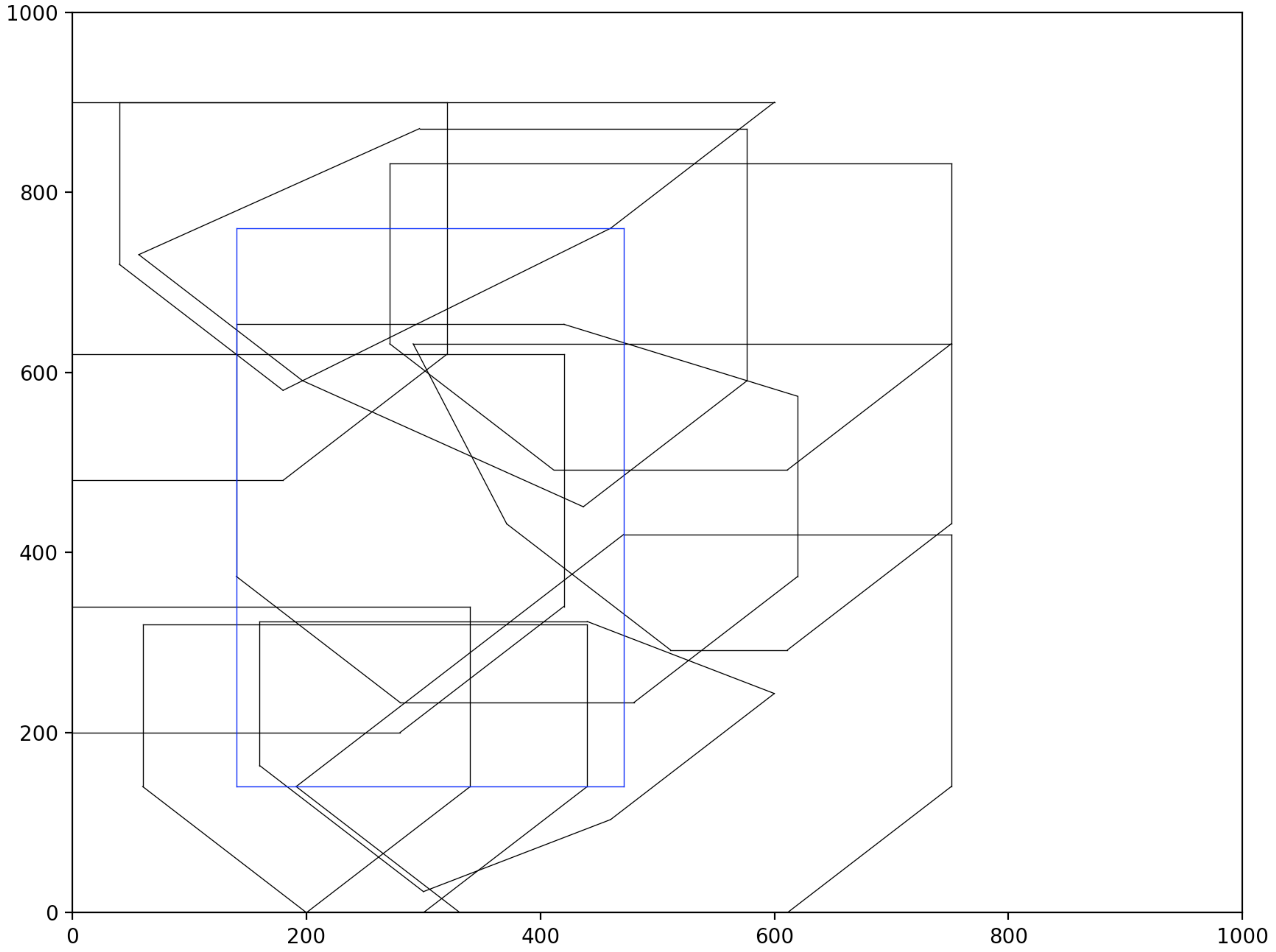
需要在蓝色区域内寻找某个位置，该位置到包含该位置的所有多边形的最短距离之和最短

算法局限性原因分析

现阶段无法进行全局检索的原因是，研究人员认为没有办法对蓝色区域内点进行枚举[10]

但是其实可以简单证明，潜在最佳位置是可以枚举的 -»

说明：下图为检索过程的NFP(黑色)与IFR(蓝色)的截图，参考点必在IFR(Inner-fit polygon)蓝色多边形的内部



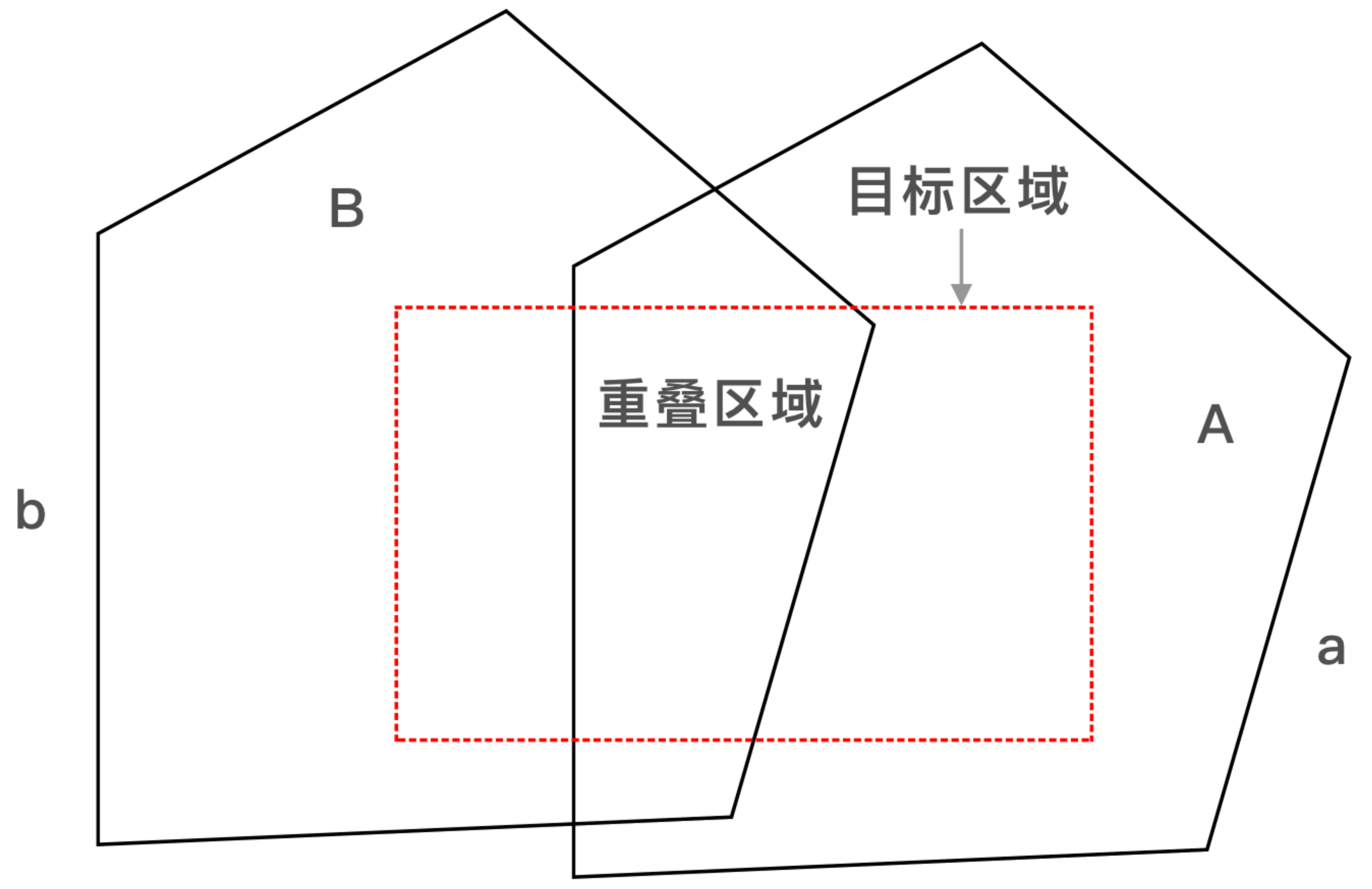
凸多边形情况下如何求解目标位置?

Target position in case of convex polygon

基础定理

定理：凸多边形内任画某区域，该区域到多边形边界距离最短的位置，必为该区域顶点或边上的点

证明：该定理可以很简单用线性规划的性质证明，具体参考P12



延伸推导

满足下述约束，且Target Region为凸集时，(x,y)必为Target Region目标区域与NFP的交点、NFP之间的交点、或目标区域和NFP的顶点

说明：PD 表示该位置的Penetration Depth，即该位置到NFP边界的最短距离。

$$Minimize \quad f(x,y) = \sum_{i=0}^{i=n} PD_i$$

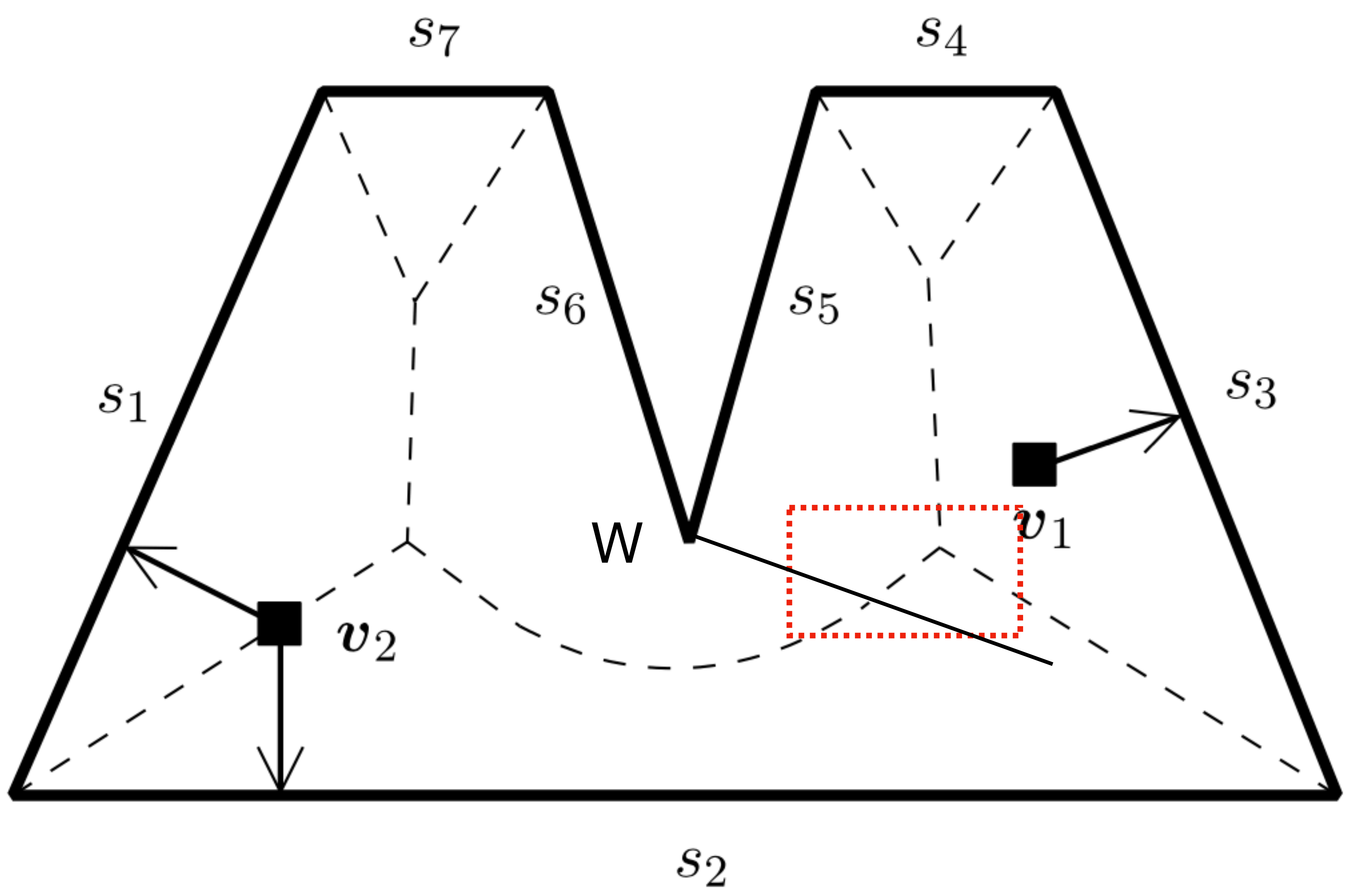
$$Subject\ to \quad (x,y) \in Target\ Region$$

$$PD_i = Min\{distance_{(x,y),NFP_i}\}$$

分析：将基础定理应用到多个重合时是一致的，这里不给出细致证明。但是，基于该原理，我们可以只对有限目标点检索，即可获得重叠最小的位置。

PD算法修正及凹多边形检索目标

How to modify pd algorithm in non-convex case



凹多边形中轴线

中轴线(Medial axis)将形状划分成不同的区域，比如V1的PD渗透深度必然为其到S3[线段]的距离。但是在凹点W所对应的区域，其PD渗透深度可能为其到S6/S5或凹点W的距离。

凹多边形的差异性

1. 红色区域内到边界距离最短的位置，可能是到凹点W的距离最短
2. 红色区域内对边做垂线，若垂足在延长线上，可能该直线不通过其他边，导致上页的定理不成立

PD修正方式

将点到点的距离，由欧式距离修改为曼哈顿距离，比如某点到w(x0,y0)的距离用下述公式，为分阶段一次线性函数

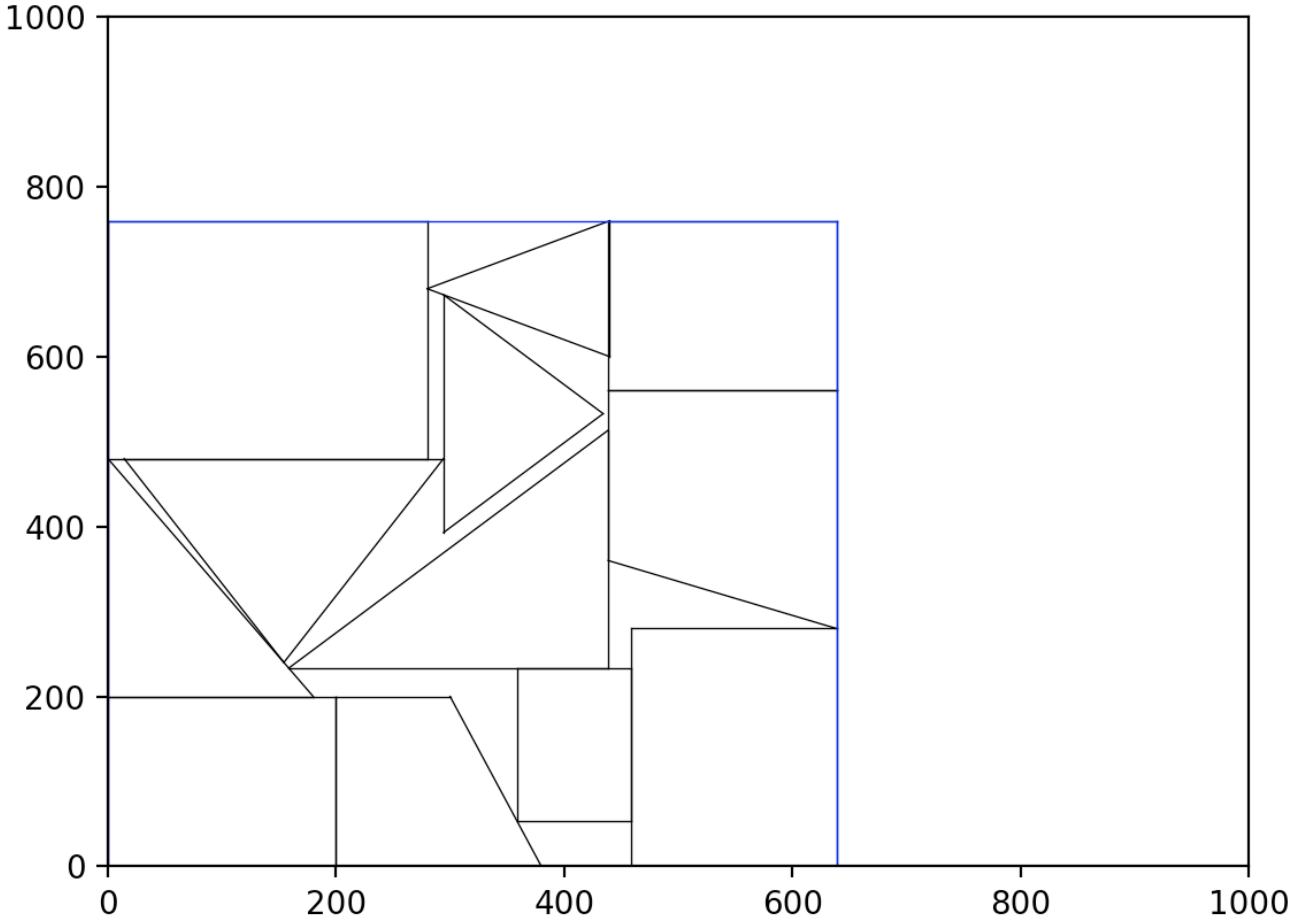
$$PD(x, y) = |x - x_0| + |y - y_0|$$

检索目标修正

1. 由于垂线问题，需要将顶点的垂线与目标区域的交点纳入潜在最优位置[证明]
2. 求解PD过程中，需要判断垂足是否在线段上

实验情况

Experimental Results



最佳结果案例

当前主要在只有凸多边形的数据集Fu和Dighe上进行了测试，Fu的最佳排样效果如上

实验结果说明与分析

1. fu的最佳利用结果90%±0.5%，接近最佳结果

凸集理论上可以检索到最优位置，在观察数据中我们发现也不存在因为全局检索陷入局部最优情况，所以理论上可以达到最优结果，仍在分析原因。

2. 相对于布谷鸟检索，需要检索的位置下降了80%+

检索点的数目随形状数目增加是线性增长，PD的计算速度也能提升，主要时间将花费在判断NFP重叠，理论分析，该方案可以确实提高检索效率。

现阶段研究情况

凸集的利用率无法提升原因正在分析，可能是几何计算问题；凸集的C++版本仍然测试，暂时不考虑差集计算效率；凹集划分实验正在落实，相对复杂，理论上可以实现较好的效果。

文献综述

排样问题的几何原理、概述以及分类

[1] Bennell J A, Oliveira J F. The geometry of nesting problems: A tutorial[J]. European Journal of Operational Research, 2008, 184(2): 397-415.

[2] Bennell J A, Oliveira J F. A tutorial in irregular shape packing problems[J]. Journal of the Operational Research Society, 2009, 60(1): S93-S105.

[3] Wäscher G, Haußner H, Schumann H. An improved typology of cutting and packing problems[J]. European journal of operational research, 2007, 183(3): 1109-1130.

NFP求解算法

常用旋转求解方案进行求解，或者为明可夫斯基和进行求解

[4] Burke E K, Hellier R S R, Kendall G, et al. Complete and robust no-fit polygon generation for the irregular stock cutting problem[J]. European Journal of Operational Research, 2007, 179(1): 27-49.

初始解获得方案

采用左底部算法或者是重心最低算法获取初始解

[5] Liu H, He Y. Algorithm for 2D irregular-shaped nesting problem based on the NFP algorithm and lowest-gravity-center principle[J]. Journal of Zhejiang University-Science A, 2006, 7(4): 570-576.

[6] Dowsland K A, Vaid S, Dowsland W B. An algorithm for polygon placement using a bottom-left strategy[J]. European Journal of Operational Research, 2002, 141(2): 371-381.

基于Layout的检索优化

通过为形状寻找更优位置，达到优化整体布局的结果

[7] Elkeran A. A new approach for sheet nesting problem using guided cuckoo search and pairwise clustering[J]. European Journal of Operational Research, 2013, 231(3): 757-769.

[8] Imamichi T, Yagiura M, Nagamochi H. An iterated local search algorithm based on nonlinear programming for the irregular strip packing problem[J]. Discrete Optimization, 2009, 6(4): 345-361.

[9] Egeblad J, Nielsen B K, Odgaard A. Fast neighborhood search for two-and three-dimensional nesting problems[J]. European Journal of Operational Research, 2007, 183(3): 1249-1266.

[10] Leung S C H, Lin Y, Zhang D. Extended local search algorithm based on nonlinear programming for two-dimensional irregular strip packing problem[J]. Computers & Operations Research, 2012, 39(3): 678-686.

[11] Umetani S, Yagiura M, Imahori S, et al. Solving the irregular strip packing problem via guided local search for overlap minimization[J]. International Transactions in Operational Research, 2009, 16(6): 661-683.

基于Layout的线性规划模型

通过限制形状相对位置，实现建立全局线性规划模型，进行持续优化

[12] Gomes A M, Oliveira J F. Solving irregular strip packing problems by hybridising simulated annealing and linear programming[J]. European Journal of Operational Research, 2006, 171(3): 811-829.

[13] Bennell J A, Dowsland K A. Hybridising tabu search with optimization techniques for irregular stock cutting[J]. Management Science, 2001, 47(8): 1160-1172.

几何概念图示

Geometry concept

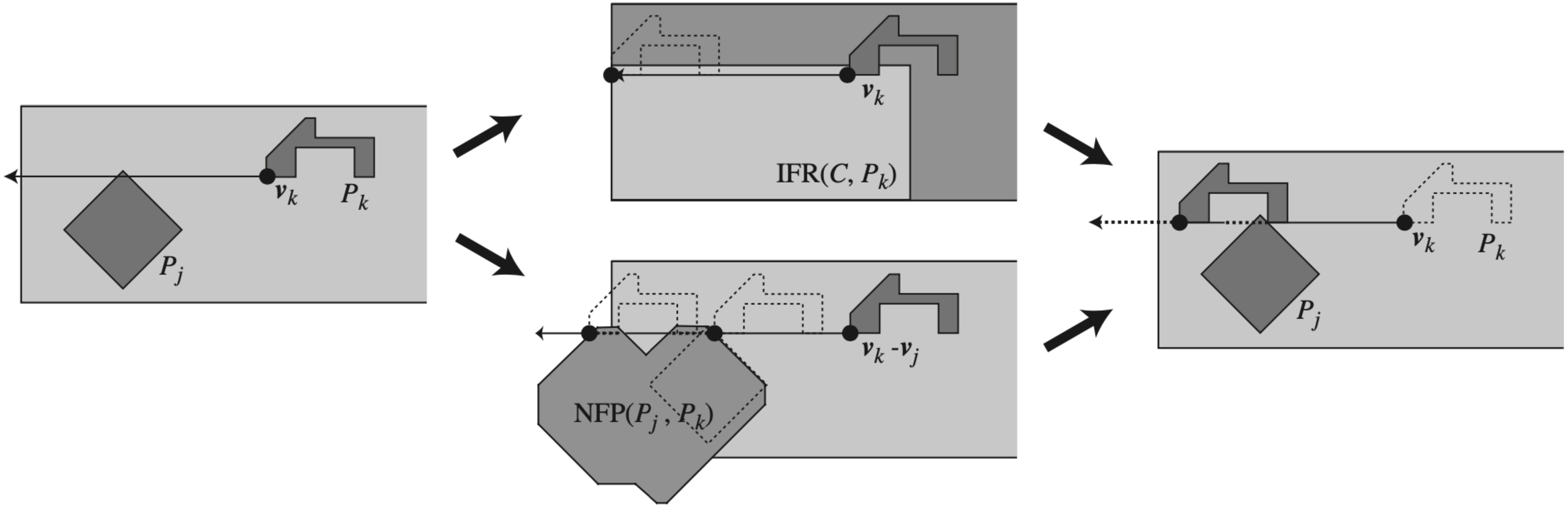


Fig. 5. Finding the left-most feasible position of a polygon P_k.

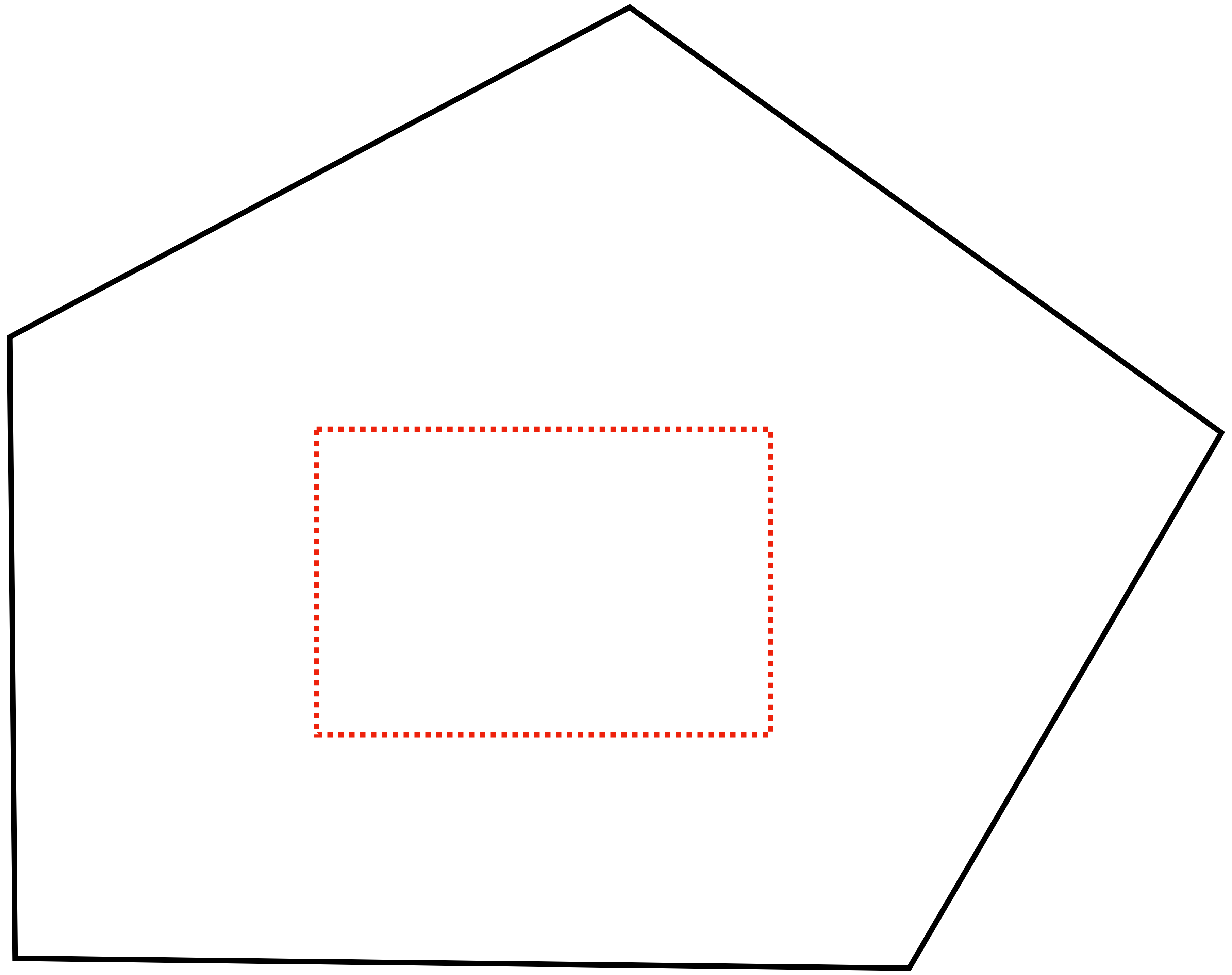
几个重要的几何概念图示

IFR是多边形在容器内绕着容器边界运动时，参考点形成的轨迹，只有参考点在IFR内形状才在容器内；NFP为某形状绕另一个形状旋转，参考点形成的轨迹，只有参考点在NFP外两个形状才没有重叠

Use modified penetration depth and guided search to solve nesting problem

定理证明

Proof in the case of convex polygons



定理：凸多边形内任画某区域，该区域到多边形边界距离最短的位置，必有该区域顶点

证明过程

Step 1 点(x,y)到某条直线的距离f(x,y)为与x,y相关的线性函数，可以写为f(x,y)=ax+by+c，a/b/c为常数

Step 2 由凸多边形的性质可知，凸多边形内任取一点，对每条边做垂线，该垂线的垂足必在该边的线段上（记为Edge1）或通过另一条边的线段（记为Edge2），后一种情况中，该点到Edge2的距离小于其到Edge1的距离

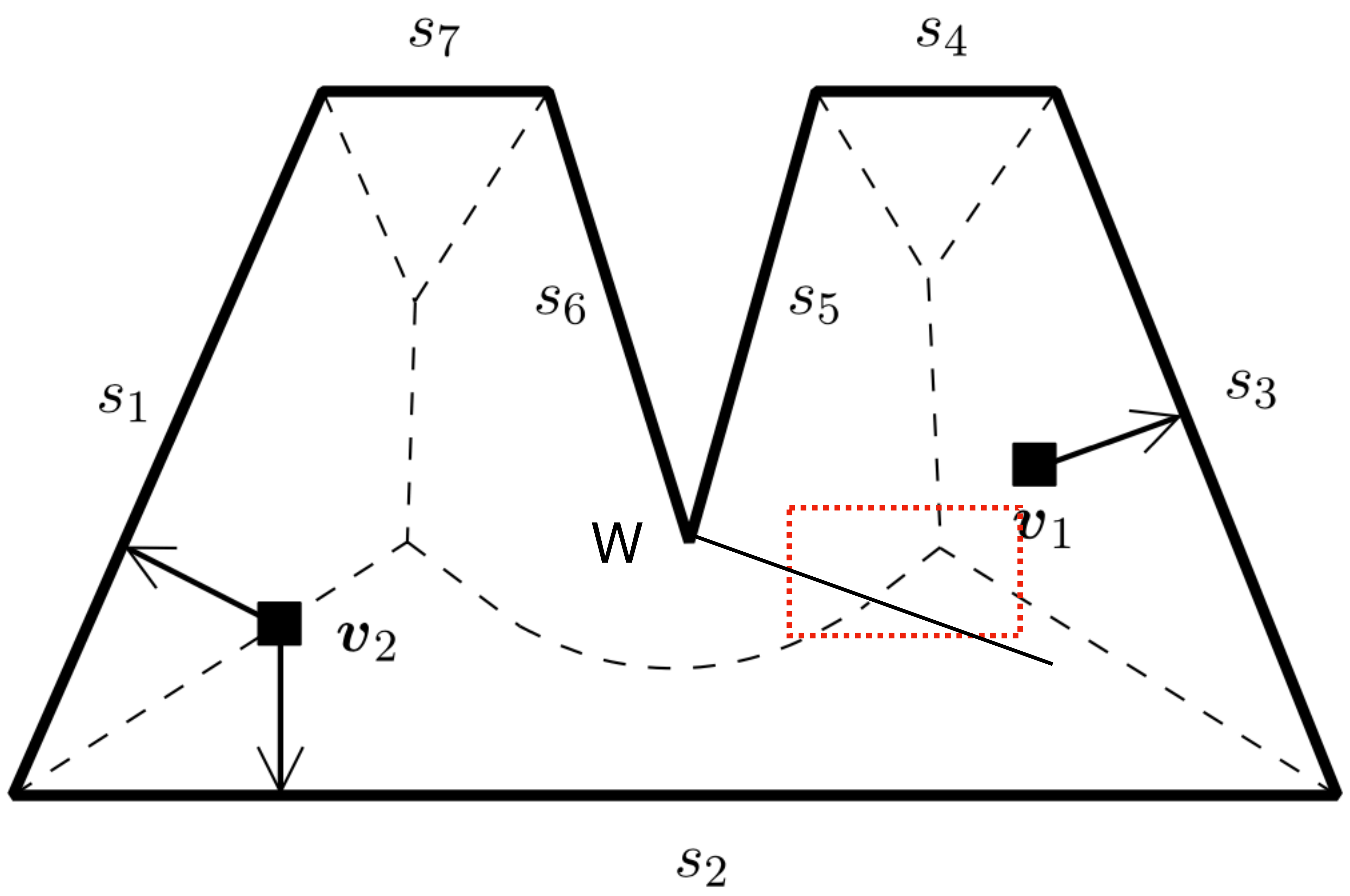
Step 3 由上一条可知，该点到多边形边界的最短距离必为其与某一条边的垂线的长度，即到该边/线段的距离

Step 4 由线性规划原理可知，二维线性目标函数在某区域内的最值，必在该边界的顶点或边界，所以任意区域到每条边的最短距离的位置必然在该区域的边界

Step 5 由Step3/4可知，该区域内到边界的最短距离的位置必然有该区域的顶点

证明凹多边形最优点情况

Proof non-convex case



简要证明：最优点可能在某条边顶点的垂线与目标区域的交点

由于只有当某点的垂足在线段上时，朝该垂足方向平移点到直线的距离，能够去除重叠的距离

只有在垂线某一侧时，点到直线的距离才有计算PD渗透深度的意义，即是一个关于x,y的线性函数

所以，最优位置可能在某条边顶点的垂线与目标区域的交点上

优化过程

Optimization Procedure

$(\boldsymbol{v}_1, \boldsymbol{o}_1) = GenerateInitialSolution(\mathcal{P}, \mathcal{O}, W)$ // *Section* 6
$(\boldsymbol{v}_{best}, \boldsymbol{o}_{best}) = (\boldsymbol{v}_1, \boldsymbol{o}_1)$
$(\boldsymbol{v}_{cur}, \boldsymbol{o}_{cur}) = (\boldsymbol{v}_1, \boldsymbol{o}_1)$
$L_{best} = L(\boldsymbol{v}_1, \boldsymbol{o}_1)$ // *Using Eq.* (3)
$L = (1 - r_{dec})\, L_{best}$
while time limit **do**
  $(\boldsymbol{v}_{cur}, \boldsymbol{o}_{cur}) = MinimizeOverlap(\mathcal{P}, \mathcal{O}, W, L, \boldsymbol{v}_{cur}, \boldsymbol{o}_{cur})$
  if $(\boldsymbol{v}_{cur}, \boldsymbol{o}_{cur})$ is feasible **then**
    $(\boldsymbol{v}_{best}, \boldsymbol{o}_{best}) = (\boldsymbol{v}_{cur}, \boldsymbol{o}_{cur})$
    $L_{best} = L$
    $L = (1 - r_{dec}) L_{best}$
  else
    $L = (1 + r_{inc}) L_{best}$
  end if
end while
return $(\boldsymbol{v}_{best}, \boldsymbol{o}_{best}, L_{best})$

Algorithm 5. $MinimizeOverlap(\mathcal{P}, \mathcal{O}, W, L, v, o)$

$n = size(\mathcal{P})$
Initialize penalty weights μ_{ij} to 1.0
$it = 0$
$Fitness = +\infty$
while $it < N_{mo}$ **do**
  Generate random permutation Q of set $S = \{1, 2, \ldots, n\}$
  for $i = 1$ *to* n **do**
    $F = F'(v_i, o_i)$ // *Using* Eq. (18)
    for each $o \in O_i$ **do**
      $v'_i = CuckooSearch(P_{Q_i}, o, W, L)$ // *Find new position*
      if $F'(\boldsymbol{v}'_i, o) < F$ **then**
        $F = F'(v'_i, o)$
        $(v_i, o_i) = (v'_i, o)$
      end if
    end for
  end for
  $F = F(\boldsymbol{v}, \boldsymbol{o})$ // *Using* Eq. (5)
  if $F = 0$ **then**
    return $(\boldsymbol{v}, \boldsymbol{o})$ // *The solution is feasible*
  else if $F < Fitness$ **then**
    $Fitness = F$
    $it = 0$
  end if
  Increase penalty weights μ_{ij} according to rule (19)
  $it = it + 1$
end while
return $(\boldsymbol{v}, \boldsymbol{o})$

Use modified penetration depth and guided search to solve nesting problem