## 第三章 航迹融合

3.1背景介绍

在上一章中，将同一船舶目标的电子侦察数据和AIS数据关联起来。在本章，为了得到船舶目标的更准确表征，先对电子侦察和AIS的实时测量值进行加权融合，选择最优的权重系数，将电子侦察和AIS数据点融合为一个数据点，然后再对融合后的航迹进行进一步处理，使用卡尔曼滤波算法进行光滑，进一步减小误差。

3.2航迹融合研究现状综述

3.3基于航迹动态质量和静态传感器精度的航迹融合算法

基于传感器精度确定权值原理

电子侦察和AIS作为不同传感器从不同位置对同一船舶目标进行检测，其中第i个传感器的测量方程为： 其中为船舶目标参数真值， 为测

量值， 为测量误差。测量误差满足条件： ;各个传感器之前测量误差是相互独立的即。

最优权值的选择就是按照一定的准则，从测量数据 用权值乘以每个传感器的测量值，其目的是有效利用每个传感器的测量值，降低测量值的误差造成的精度损失。下面讨论如何确定最优权值的算法。

首先建立融合后估计值的表达式：，其中 为第i个传感器的权值系数，要求是无偏估计必须满足如下的条件：

+

即当时 是 的无偏估计

其次，建立权值估计误均方差表达式：

= 0

= = min

利用拉格朗日条件求极值，可以确定使得 最小的权重系数 (i=1,2,······n)求解过程如下：

=

建立方程组：

求解该方程组可到

可得：

可知加权均方误差为：

由公式可知，如果传感器误差方差小，则其权值系数大，在最后融合数据中占比大，反之，传感器误差方差大，则其权值系数小，在最后融合数据中占比小。这样可以避免盲目的平均使用各个传感器的测量值。由公式 可知采用最优权值法其均方误差比任意单个传感器的数据误差要小，这说明多个精度一般的传感器测量数据经过融合后，精度能得到有效的提升，这也是本研究做航迹融合的最重要的原因。

上面基于传感器静态精度的方法在理想情况下是得到最优权值，由于 这两个重要的假设在实际情况中不一定满足，对于AIS数据来说，由于本质是电磁波的传播，容易受到地形、天气、建筑物障碍等影响。对于电子侦察数据来说，由于是卫星采集船舶上雷达波、除了地形、天气。障碍物的影响，还受到卫星运行轨道，卫星相对位置的影响。即实际环境中公式1 2没法满足。基于此本文提出基于动态航迹质量和静态传感器误差的融合算法。

动态航迹点质量：在时刻t对于电子侦察数据的航迹点和AIS数据的航迹点的质量评估，算法是一个动态迭代的过程，在t时刻，t-1时刻的航迹融合已经完成，t-1时刻的航迹质量定义为该时刻某一传感器的测量航迹点与融合后航迹点的相似程度。在采样间隔较小的情况下，可以用t-1时刻的航迹质量近似替代t时刻的航迹质量。所以，可以将动态航迹点质量的计算转化为传感器测量航迹点与融合后航迹点之间相似度计算

航迹点之间相似度计算：

航迹点相似度计算基于航迹点之间的距离计算，对于航迹点向量的维度是经度、纬度、航向、航速。经度和纬度属性可以用来计算位置点之间的距离，而COG和SOG由于并不是独立于经纬度特征的，实际传感器上所得到的的COG和SOG与经纬度位置信息有很大联系。因此，在相应的距离计算时，不易使用传统的将向量的各个维度视为独立变量进行计算，在航迹点相似度计算中首先最重要的距离是位置距离，当位置距离超过一定阈值，航向和航速之间的距离变得没有意义，即只用位置距离来计算航迹点之间的距离，当位置距离在阈值范围内，需要考虑航向和航速之间的距离，但是，位置、航速、航向三者在航迹距离计算中的权重是不同的，公式化表达如下：

航迹点相似度计算伪代码：

输入：

输出：航迹点相似度值

1. position\_distance = haversine()//计算两点之间的位置距离
2. *if* (position\_distance>T1)

return 1-

position\_distance =

COG\_distance = //计算原始航向距离

COG\_distance = //归一化航向距离

SOG\_distance = //计算原始航速距离

SOG\_distance= //归一化航速距离

return

使用上述航迹点相似度计算算法可以得到实时不同传感器测量数据的质量，结合静态传感器误差可以实现基于动态航迹质量和静态传感器误差的加权融合算法，同时考虑到电子侦察数据误差较大，

基于动态航迹质量和静态传感器误差的融合算法伪代码：

输入：电子侦察数据航迹 AIS航迹

电子侦察数据静态测量方差： AIS静态测量方差

静态传感器误差权值 动态航迹质量权值  其中

输出：融合后的航迹

//某一时刻电子侦察和AIS差距过大，取AIS作为输出结果

1. *if*( t=1 ) //第一个航迹点，没有动态航迹质量部分，按照静态测量误差融合

2 + //静态融合

3.//计算航迹质量

4.

3. for t=2 to n

4 = + +

//更新航迹质量

add

return

3.4基于卡尔曼滤波的航迹平滑算法

3.5实验验证

3.6本章小结