Додаток А

Список публікацій здобувача за темою дисертації та відомості про апробацію результатів дисертації

Список публікацій, в яких опубліковані основні наукові результати дисертації:

1. Barannik, D., Bekirov, A., Korotin, Barannik, V., S., Veselska, O., Wieclaw, L. (2020). Method of Safety of Informational Resources Utilizing the Indirect Steganography. In: Zawiślak, S., Rysiński, J. (eds) Engineer of the XXI Century. EngineerXXI 2018. Mechanisms and Machine Science, vol 70. *Monogrpah,* Springer, Cham. https://doi.org/10.1007/978-3-030-13321-4_17. **Scopus**

2. Barannik D., Babenko Y., Hahanova A., Karpiinski M., Shaikhanova A., Shmakov V., Smailova B., Veselska O., Yroshenko V. The Technology of Structural Classification of Video Frames in Intelligent Info-Communication Systems. "*Development of technology analys for the content semantics,"* in Engineer of XXI Century - We Design the Future, Bielsko-Biała, Poland: ATH, 2020. P.31 – 40. *Monogrpah,* Wydawnictwo Naukowe Akademii Techniczno-Humanistycznej w Bielsku-Białej

3. Бараннік Д.В., Бекіров А.Е., Бараннік В.В. Основы теории структурно-комбинаторного стеганографического кодирования: монография. – Харків.: Видавництво «Лідер», 2017. – 256 с.

4. Баранник Д.В. Анализ возможности использования маскирования при выявлении областей для стеганографического встраивания // Наукоемкие технологии в инфокоммуникациях: обработка информации, кибербезопасность, информационная борьба: коллективная монография [под редакцией В.В. Баранника, В.М.Безрука]. – Х.: ТОВ «Видвництво « Лідер»», 2017. – С. 375 - 389.

5. Dmitriy Barannik, Serhii Sidchenko, Sergii Shulgin, Vladimir Barannik, Valeriy Barannik, Anton Datsun (2021). Devising a conceptual method for

generating cryptocompression codograms of images without loss of information quality. Eastern-European Journal of Enterprise Technologies, 4 (2 (112)), 6–16. doi: https://doi.org/10.15587/1729-4061.2021.237359. **Scopus Квантіль Q 3**

6. Barannik D., Shatun O., Kobtseva V., Barannik V. Development of an indirect method of steganographic data hiding in the container image contour // Informatyka Automatyka Pomiary w Gospodarce i Ochoronie Środowiska (IAPGOŚ). – 2018. - Volume 8.– №4. – pp. 22 – 25. DOI: 10.5604/01.3001.0012.8016. (***стаття опублікована в журналі Європейського Союзу***)

7. Бараннік Д.В. Метод стеганокомпресійного декодування інформації для бездротових інфокомунікаційних технологій // Наукоємні технології. – 2024. - №1. – С. 29 – 35. DOI: https://doi.org/10.18372/2310-5461.61.18512.

8. Бараннік Д.В., Оніщенко Р.С., Ревва К.В., Бараннік В.В., Бабенко Ю.М. Метод оцінювання ефективності усічено-позиційного кодування для інфокомунікаційних систем // Наукоємні технології. – 2024. - №2. – С. 175-184. doi: 10.18372/2310-5461.62.18711.

9. Бараннік Д.В. Метод двокаскадної імплантації прихованої інформації на основі стеганокомпресійних перетворень // Інформаційні технології та електронна інженерія. – 2024. - №1. – С. 31 – 38. DOI: https://doi.org/10.23939/ictee2024.01.031

10. Бараннік Д.В. Технологія стеганографічних перетворень в поліадичному базисі відеосегментів // Сучасна спеціальна техніка. – 2023. - №4. – С. 17 – 24.

11. Бараннік Д.В. Технологія приховування інформативного контенту в динамічному потоці відеосегментів // Наукоємні технології. – 2023. - №4. – С. 408-415. DOI: https://doi.org/10.18372/2310-5461.60.18270

12. Бараннік Д.В. Методологія створення стеганокомпресійних систем на основі поліадичного базису // Сучасна спеціальна техніка. – 2023. - №3. – С. 25 – 35. doi.org/10.36486/mst2411-3816.2023.3.(74)

13. Бараннік Д.В. Метод стеганокомпресійного кодування на основі поліадичного базису // Наукоємні технології. – 2023. - №3. – С. 271-279. DOI: https://doi.org/10.18372/2310-5461.59.17948

14. Бараннік Д.В., Колесник В.О., Бараннік В.В., Шульгін С.С., Заїчко К.В. Метод виявлення та обробки значимої інформації в просторово-часовій області відеознімку // Сучасна спеціальна техніка. – 2022. - №3. – С. 20 – 31.

15. Бараннік Д.В., Твердохліб В.В., Колесник В.О., Бараннік В.В., Бараннік Н.В., Онищенко Р.С. Метод стеганографічного захисту даних з урахуванням особливостей змісту фрагментів контейнерів графічного типу // Сучасна спеціальна техніка. – 2021. - №4. – С. 25 – 33.

16. Баранник Д.В., Красноруцкий А.А., Шульгин С.С., Хаханова А.В.,. Метод определения сильно информативных сегментов аэрофотоснимка // Радіоелектронні і комп’ютерні системи. – 2018. - №1, pp. 15-22. DOI: https://doi.org/10.32620/reks.2018.1.02

17. Баранник Д.В., Шатун О.Н., Баранник В.В. Метод непрямого стеганографічного вбудовування даних в зображення-контейнер з урахуванням інформації контуру // Наукоємні технології. – 2018. – №2. – С. 232-239. DOI: https://doi.org/10.18372/2310-5461.38.12828

18. Бараннік Д. В., Сорокун А. Д., Дворський М. В., Бараннік В. В. Вдосконалення методів компенсації руху динамічних об’єктів у відеопотоці в системі відеоконференцзв’язку // Наукоємні технології. – 2018. - № 3. – С. 294-300.

19. Бараннік Д.В., Сидченко С.А. Метод криптосемантичного представлення зображень на основі плаваючої схеми системи поліадичного кодування в диференціальному базисі // Наукоємні технології. – 2017. - №1. – С. 46 – 52.

20. Бараннік Д.В., Тарнополов Р.В., Бараннік В.В., Хаханова Г.В. Метод підвищення оперативності та конфіденційності відеоданих в інфокомунікаційних технологіях // Збірник наукових праць Харківського університету Повітряних Сил. – 2017. - №4. – С. 59-61.

21. Баранник Д.В., Бекиров А.Э. Концепция структурного стеганографического кодирования с маскированием // АСУ та прилади автоматики. - 2014. - Вип.168. - С. 4 - 11

22. Баранник Д.В., Бекиров А.Э., Баранник В.В. Стеганографическая система на основе неравновесного позиционного кодирования // Радіоелектроніка та інформатика. - 2014. - №4. - С. 39 – 45.

23. Патент на корисну модель № 144599, Україна, МПК H03M 7/30, H03M 13/00 Спосіб нерівноваговoго позиційного кодування для зменшення інформаційної інтенсивності відеопотоку / Бараннік Д.В., Хіменко В.В., Куліца О.С., Твердохліб В.В., Фустій В.С. та ін. - № u202002968; заяв. 13.10.2020; опубл. 12.10.2020; Бюл. № 19. - 9 с.

Список публікацій, які засвідчують апробацію матеріалів дисертації:

1. D. Barannik, "Method for Hiding Informative Content in a Dynamic Stream of Segments," 2024 IEEE 17th International Conference on Advanced Trends in Radioelectronics, Telecommunications and Computer Engineering (TCSET), Lviv, Ukraine, 2024, pp. 1-5, doi: 10.1109/TCSET64720.2024.10755687.

2. Barannik D., Ustymenko F., Hurzhii P., Barannik V., Manakov V. and Tsimura Y. "Method of Selective Video Segment Processing for Intelligent Video Image Quality Enhancement Technologies," 2023 IEEE 5th International Conference on Advanced Information and Communication Technologies (AICT), Lviv, Ukraine, 2023, pp. 217-220, doi: 10.1109/AICT61584.2023.10452422. **Scopus**

3. Barannik D., Barannik V. “Steganographic Coding Technology for Hiding Information in Infocommunication Systems of Critical Infrastructure”, 2022 IEEE 4th International Conference on Advanced Trends in Information Theory (ATIT), Kyiv, Ukraine, 2022, pp. 88-91. DOI: 10.1109/ATIT58178.2022.10024185. **Scopus**

4. Barannik D., Tverdokhlib V., Kulitsa O., Chornomaz I., Savchuk M. and Parkhomenko M. "Method of Selective Steganographic Data Hiding Based on Graphic Containers," 2022 IEEE 16th International Conference on Advanced Trends in

Radioelectronics, Telecommunications and Computer Engineering (TCSET), Lviv-Slavske, Ukraine, 2022, pp. 499-502, doi: 10.1109/TCSET55632.2022.9767014. **Scopus**

5. Barannik V., Barannik D., Sidchenko Y., Ignatyev O., Slobodyanyuk O. and Lazuta R., "Method of Masking Information in the Contours of Video Images," 2021 IEEE 4th International Conference on Advanced Information and Communication Technologies (AICT), Lviv, Ukraine, 2021, pp. 282-287, doi: 10.1109/AICT52120.2021.9628973. **Scopus**

6. Barannik D., Himenko V., Turenko S., Barannik N., Ignatyev O. and Matviichuk-Yudina O. "Model of Steganographic System Depending on Indirect Conditional Dependencies," 2021 IEEE 3rd International Conference on Advanced Trends in Information Theory (ATIT), Kyiv, Ukraine, 2021, pp. 36-39, doi: 10.1109/ATIT54053.2021.9678547. **Scopus**

7. Barannik D. "Stegano-Compression Coding in a Non-Equalible Positional Base," 2020 IEEE 2nd International Conference on Advanced Trends in Information Theory (ATIT), Kyiv, Ukraine, 2020, pp. 83-86, doi: 10.1109/ATIT50783.2020.9349328. **Scopus**

8. Barannik D., Barannik N. Barannik V. and Ryabukha Y. "Indirect Steganographic Embedding Method Based on Modifications of the Basis of the Polyadic System," 2020 IEEE 15th International Conference on Advanced Trends in Radioelectronics, Telecommunications and Computer Engineering (TCSET), Lviv-Slavske, Ukraine, 2020, pp. 699-702, doi: 10.1109/TCSET49122.2020.235522. **Scopus**

9. Barannik D., Fustii, V., Tverdokhleb V., Slobodyanyuk O., Havrylov D. and Shevchenko I. "Evaluation the Potential Performance of the DCT-Transformants Non-Equilibrium Positional Encoding Method," 2019 IEEE International Conference on Advanced Trends in Information Theory (ATIT), Kyiv, Ukraine, 2019, pp. 41-44, doi: 10.1109/ATIT49449.2019.9030460. **Scopus**

10. Barannik D., Fustii V. Barannik V., and Parkhomenko M. "Evaluation of Effectiveness of Masking Methods of Aerial Photographs," 2019 3rd

International Conference on Advanced Information and Communications Technologies (AICT), Lviv, Ukraine, 2019, pp. 415-418, doi: 10.1109/AIACT.2019.8847820. **Scopus**

11. Barannik D., Shatun O., Dodukh O., Ryabukha Y. and Tverdokhleb V. "The Indirect Method of Steganographic Embedding of Data in an Image Container Based on the Information of the Contour," 2018 International Scientific-Practical Conference Problems of Infocommunications. Science and Technology (PIC S&T), Kharkiv, Ukraine, 2018, pp. 490-494, doi: 10.1109/INFOCOMMST.2018.8632155. **Scopus**

12. Barannik D., Bekirov A., Lekakh A. and Barannik V. "A steganographic method based on the modification of regions of the image with different saturation," 2018 14th International Conference on Advanced Trends in Radioelecrtronics, Telecommunications and Computer Engineering (TCSET), Lviv-Slavske, Ukraine, 2018, pp. 542-545, doi: 10.1109/TCSET.2018.8336260. **Scopus**

13. Barannik D. Alimpiev A., Bekirov A., Barannik V. and Barannik N. "Detections of sustainable areas for steganographic embedding," 2017 IEEE East-West Design & Test Symposium (EWDTS), Novi Sad, Serbia, 2017, pp. 1-4, doi: 10.1109/EWDTS.2017.8110028. **Scopus**

14. Barannik D., Suprun O., Bekirov A., Frolov O. “The new method of secure data transmission on the indirect steganography basis”, 2016 IEEE East-West Design & Test Symposium (EWDTS), Yerevan, Armenia, 2016, pp. 1-4. DOI: 10.1109/EWDTS.2016.7807754. **Scopus**

Апробація результатів дисертації була проведена на:

2024 IEEE 5th International Conference on Advanced Trends in Information Theory (ATIT), Lviv, Ukraine;

2024 IEEE 17th International Conference on Advanced Trends in Radioelectronics, Telecommunications and Computer Engineering (TCSET), Lviv, Ukraine*;*

2023 IEEE 5th International Conference on Advanced Information and Communication Technologies (AICT), Lviv, Ukraine;

“Development of technology analys for the content semantics,” in Engineer of XXI Century - We Design the Future, Bielsko-Biała, Poland: ATH;

2022 IEEE 4th International Conference on Advanced Trends in Information Theory (ATIT), Kyiv, Ukraine;

2022 IEEE 16th International Conference on Advanced Trends in Radioelectronics, Telecommunications and Computer Engineering (TCSET), Lviv-Slavske, Ukraine;

2021 IEEE 4th International Conference on Advanced Information and Communication Technologies (AICT), Lviv, Ukraine;

2021 IEEE 3rd International Conference on Advanced Trends in Information Theory (ATIT), Kyiv, Ukraine;

2020 IEEE 2nd International Conference on Advanced Trends in Information Theory (ATIT), Kyiv, Ukraine;

2020 IEEE 15th International Conference on Advanced Trends in Radioelectronics, Telecommunications and Computer Engineering (TCSET), Lviv-Slavske, Ukraine;

2019 IEEE International Conference on Advanced Trends in Information Theory (ATIT), Kyiv, Ukraine;

2019 3rd International Conference on Advanced Information and Communications Technologies (AICT), Lviv, Ukraine;

2018 International Scientific-Practical Conference Problems of Infocommunications. Science and Technology (PIC S&T), Kharkiv, Ukraine;

2018 14th International Conference on Advanced Trends in Radioelecrtronics, Telecommunications and Computer Engineering (TCSET), Lviv-Slavske, Ukraine;

2017 IEEE East-West Design & Test Symposium (EWDTS), Novi Sad, Serbia;

2016 IEEE East-West Design & Test Symposium (EWDTS), Yerevan, Armenia;

Додаток Б

Фрагмент програмної реалізації створених методів стеганокомпресійного кодування

```
class MainProcessor2(
   private val modules: List<IDataProcessor2<ProcessingState,
ProcessingState>>
) : IDataProcessor2<ProcessingState, ProcessingState> {

   constructor(
       params: Params
   ) : this(params.moduleParams.map { ProcessorModuleFactory.getModule2(it)
})

   enum class Mode { DIRECT, REVERSE }

   data class Params(val moduleParams:
List<ProcessorModuleFactory.ModuleParams>)

   fun run(data: ProcessingState, mode: Mode = Mode.DIRECT): ProcessingState
{
       TimeCalculatorUtils.startNewTrack("start $mode: ${data.fileOfBar},")
       var tmp = data
       val functionToCall = when(mode) {
           Mode.DIRECT -> IDataProcessor2<ProcessingState,
ProcessingState>::direct
           Mode.REVERSE -> IDataProcessor2<ProcessingState,
ProcessingState>::reverse
       }
       when (mode) {
           Mode.DIRECT -> modules
           Mode.REVERSE -> modules.asReversed()
       }.forEach {
           Log.d("RUN MODULE: $it")
           tmp = functionToCall.call(it, tmp)
           TimeCalculatorUtils.append(it.toString())
       }
       Log.d("TimeCalculatorUtils = " + TimeCalculatorUtils.getInfoInSec())
       Log.d("TimeCalculatorUtils = " +
TimeCalculatorUtils.getInfoInPercent())
       return tmp
   }

   override fun direct(data: ProcessingState): ProcessingState {
       return run(data, Mode.DIRECT)
   }

   override fun reverse(data: ProcessingState): ProcessingState {
       return run(data, Mode.REVERSE)
   }
}
```

IDataProcessor.kt

```
interface IDataProcessor2<D, R> {
   fun direct(data: D): R
   fun reverse(data: R): D
}
```

ProcessorModuleFactory.kt

```
object ProcessorModuleFactory {
```

```
    fun getModule2(params: ModuleParams): IDataProcessor2<ProcessingState,
ProcessingState> = when (params) {
        is ModuleParams.ReadImage -> ModuleReadRgb(params.params)
        is ModuleParams.Grey16 -> ModuleGrey16(params.params)
        is ModuleParams.RgbToYbr -> ModuleRgbToYCbCr(params.params)
        is ModuleParams.Dct -> ModuleDct(params.params)
        is ModuleParams.DumpFrame -> ModuleDumpFrame(params.params)
        is ModuleParams.Subsampling -> ModuleSubsampling(params.params)
        is ModuleParams.Opc -> ModuleOpc2(params.params)
        is ModuleParams.Opc3 -> ModuleOpc3(params.params)
        is ModuleParams.Opc3d -> ModuleOpc3Dimensions(params.params)
        is ModuleParams.Opc3ForBases -> ModuleOpc3Bases(params.params)
        is ModuleParams.Opc3dForOpc3d -> ModuleOpc3dForOpc3d(params.params)
        is ModuleParams.Opc3ForSign -> ModuleOpc3Sign(params.params)
        is ModuleParams.OpcToBytes -> ModuleOpcToBytes(params.params)
        is ModuleParams.Opc3ToBits -> ModuleOpc3ToBits(params.params)
        is ModuleParams.Opc3dToBits -> ModuleOpc3dToBits(params.params)
        is ModuleParams.Quantization -> ModuleQuantization(params.params)
        is ModuleParams.WriteBytes -> ModuleWriteBytes(params.params)
        is ModuleParams.WriteBits -> ModuleWriteBits(params.params)
        ModuleParams.Compression -> ModuleCompressionBytes()
        is ModuleParams.Opc2Bases -> ModuleOpc2Bases(params.params)
        is ModuleParams.AnyDirection -> AnyDirectionProcessor(params.params)
        is ModuleParams.ImageAnalysis -> ModuleImageAnalysis(params.params)
//          is ModuleParams.Encryption -> ModuleEncryptionBytes(params.params)
        else -> throw Exception("module not supported $params")
    }

    sealed class ModuleParams {
        data class Grey16(val params: ModuleGrey16.Parameters =
ModuleGrey16.Parameters()) : ModuleParams()
        data class ReadImage(val params: ModuleReadRgb.Parameters =
ModuleReadRgb.Parameters()) : ModuleParams()
        data class DumpFrame(val params: ModuleDumpFrame.Parameters =
ModuleDumpFrame.Parameters()) : ModuleParams()
        data class RgbToYbr(val params: ModuleRgbToYCbCr.Parameters =
ModuleRgbToYCbCr.Parameters()) : ModuleParams()
        data class Subsampling(val params: ModuleSubsampling.Parameters =
ModuleSubsampling.Parameters()) : ModuleParams()
        data class Dct(val params: ModuleDct.Parameters =
ModuleDct.Parameters()) : ModuleParams()
        data class Quantization(val params: ModuleQuantization.Parameters =
ModuleQuantization.Parameters()) : ModuleParams()
        @Deprecated("use Opc3")
        data class Opc(val params: ModuleOpc2.Parameters =
ModuleOpc2.Parameters()) : ModuleParams()
        data class Opc3(val params: ModuleOpc3.Parameters =
ModuleOpc3.Parameters()) : ModuleParams()
        data class Opc3d(val params: ModuleOpc3Dimensions.Parameters =
ModuleOpc3Dimensions.Parameters()) : ModuleParams()
        @Deprecated("use Opc3ForBases")
        data class Opc2Bases(val params: ModuleOpc2Bases.Parameters =
ModuleOpc2Bases.Parameters()) : ModuleParams()
        data class Opc3ForBases(val params: ModuleOpc3Bases.Parameters =
ModuleOpc3Bases.Parameters()) : ModuleParams()
        data class Opc3dForOpc3d(val params: ModuleOpc3dForOpc3d.Parameters =
ModuleOpc3dForOpc3d.Parameters()) : ModuleParams()
        data class Opc3ForSign(val params: ModuleOpc3Sign.Parameters =
ModuleOpc3Sign.Parameters()) : ModuleParams()
        @Deprecated("use Opc3ToBytes")
        data class OpcToBytes(val params: ModuleOpcToBytes.Parameters =
ModuleOpcToBytes.Parameters()) : ModuleParams()
        data class Opc3ToBits(val params: ModuleOpc3ToBits.Parameters =
```

```
ModuleOpc3ToBits.Parameters()) : ModuleParams()
       data class Opc3dToBits(val params: ModuleOpc3dToBits.Params =
ModuleOpc3dToBits.Params()) : ModuleParams()
       object Compression : ModuleParams()
       data class Encryption(val params: ModuleEncryptionBytes.Parameters =
ModuleEncryptionBytes.Parameters()) : ModuleParams()
       @Deprecated("use WriteBites")
       data class WriteBytes(val params: ModuleWriteBytes.Parameters =
ModuleWriteBytes.Parameters()) : ModuleParams()
       data class WriteBits(val params: ModuleWriteBits.Parameters =
ModuleWriteBits.Parameters()) : ModuleParams()
       data class AnyDirection(val params:
AnyDirectionProcessor.Parameters<ProcessingState>) : ModuleParams()
       data class ImageAnalysis(val params: ModuleImageAnalysis.Parameters =
ModuleImageAnalysis.Parameters()) : ModuleParams()
   }
}

class ModuleRgbToYCbCr(val parameters: Parameters = Parameters()) :
   AbsDataProcessor<ProcessingData.Image, ProcessingData.Image>(
       ProcessingData.Image::class, ProcessingData.Image::class
   ), IDataProcessor2<ProcessingState, ProcessingState> {

   class Parameters

   override fun processDirectTyped(data: ProcessingData.Image):
ProcessingData.Image {
       val rgb = RGB(data.triple.first, data.triple.second,
data.triple.third)
       val yCbCr = RgbToYbrUtils.direct(rgb)
       return ProcessingData.Image(
           Triple(
               yCbCr.matrixY,
               yCbCr.matrixCb,
               yCbCr.matrixCr
           )
       )
   }

   override fun processReverseTyped(data: ProcessingData.Image):
ProcessingData.Image {
       val yCbCr = YCbCr(data.triple.first, data.triple.second,
data.triple.third)
       val rgb = RgbToYbrUtils.reverse(yCbCr)
       return ProcessingData.Image(
           Triple(
               rgb.matrixR,
               rgb.matrixG,
               rgb.matrixB
           )
       )
   }

   override fun direct(data: ProcessingState): ProcessingState {
       val rgb = RGB(data.planesOfPixel!![0], data.planesOfPixel!![1],
data.planesOfPixel!![2])
       val yCbCr = RgbToYbrUtils.direct(rgb)
       data.planesOfPixel = Triple(
           yCbCr.matrixY,
           yCbCr.matrixCb,
           yCbCr.matrixCr
       )
       return data
   }
```

```
    override fun reverse(data: ProcessingState): ProcessingState {
        val yCbCr = YCbCr(data.planesOfPixel!![0], data.planesOfPixel!![1],
data.planesOfPixel!![2])
        val rgb = RgbToYbrUtils.reverse(yCbCr)
        data.planesOfPixel = Triple(
            rgb.matrixR,
            rgb.matrixG,
            rgb.matrixB
        )
        return data
    }
}

class ModuleDct(
    val parameters: Parameters = Parameters()
) : AbsDataProcessor<ProcessingData.Image, ProcessingData.Image>(
    ProcessingData.Image::class, ProcessingData.Image::class
), IDataProcessor2<ProcessingState, ProcessingState> {

    data class Parameters(
        val asyncType: AsyncType = AsyncType.MULTI_THREAD,
        val managerParamProvider: (Int) -> DctManager.Parameters = {
DctManager.Parameters() }
    ) {
        constructor(
            processFirst: DctManager.Parameters = DctManager.Parameters(),
            processSecond: DctManager.Parameters = DctManager.Parameters(),
            processThird: DctManager.Parameters = DctManager.Parameters(),
            asyncMode: Boolean = true
        ) : this(managerParamProvider = {
            when (it) {
                0 -> processFirst
                1 -> processSecond
                2 -> processThird
                else -> throw Exception()
            }
        }, asyncType = if (asyncMode) AsyncType.MULTI_THREAD else
AsyncType.SINGLE)
    }

    private val managers = LazyMap<Int, DctManager> {
DctManager(parameters.managerParamProvider(it)) }

    private val asyncCollectionProcessor by lazy {
        val directF =
            { pair: Pair<Int, Matrix<PixelType>> -> pair.first to
managers.getOrLazy(pair.first).direct(pair.second) }
        val reverseF =
            { pair: Pair<Int, Matrix<PixelType>> -> pair.first to
managers.getOrLazy(pair.first).reverse(pair.second) }
        when (parameters.asyncType) {
            AsyncType.SINGLE ->
AsyncCollectionProcessor.SINGLE_THREAD(directF, reverseF)
            AsyncType.MULTI_THREAD ->
AsyncCollectionProcessor.MULTI_THREADS(3, directF, reverseF)
        }
    }

    override fun processDirectTyped(data: ProcessingData.Image):
ProcessingData.Image {
        return
ProcessingData.Image(Triple(direct(ProcessingState(planesOfPixel =
data.triple)).planesOfPixel!!))
```

```
    }

    override fun processReverseTyped(data: ProcessingData.Image):
ProcessingData.Image {
        return
ProcessingData.Image(Triple(reverse(ProcessingState(planesOfPixel =
data.triple)).planesOfPixel!!))
    }


    override fun direct(data: ProcessingState) = data.also {
        val pairs = it.planesOfPixel!!.mapIndexed { index: Int, ints:
Matrix<PixelType> ->
            managers.getOrLazy(index).compressionTypeMatrix =
data.planesOfCompressionType?.get(index) //todo
            index to ints
        }
        it.planesOfPixel = asyncCollectionProcessor.direct(pairs).map {
it.second }
    }

    override fun reverse(data: ProcessingState) = data.also {
        val pairs = it.planesOfPixel!!.mapIndexed { index: Int, ints:
Matrix<PixelType> ->
            managers.getOrLazy(index).compressionTypeMatrix =
data.planesOfCompressionType?.get(index) //todo
            index to ints
        }
        it.planesOfPixel = asyncCollectionProcessor.reverse(pairs).map {
it.second }
    }
}

class ModuleQuantization(
    val parameters: Parameters = Parameters()
) : AbsDataProcessor<ProcessingData.Image, ProcessingData.Image>(
    ProcessingData.Image::class, ProcessingData.Image::class
), IDataProcessor2<ProcessingState, ProcessingState> {

    data class Parameters(
        val asyncType: AsyncType = AsyncType.MULTI_THREAD,
        val managerParams: (Int) -> QuantizationManager.Parameters = {
            if (it == 0) QuantizationManager.Parameters(
                QuantizationUnit.Parameters(tableType =
QuantizationUnit.TableType.LUMINOSITY(true))
            ) else QuantizationManager.Parameters(
                QuantizationUnit.Parameters(tableType =
QuantizationUnit.TableType.CHROMATICITY(true))
            )
        }
    ) {
        @Deprecated("")
        constructor(
            first: QuantizationManager.Parameters =
QuantizationManager.Parameters(
                QuantizationUnit.Parameters(tableType =
QuantizationUnit.TableType.LUMINOSITY(true))
            ),
            second: QuantizationManager.Parameters =
QuantizationManager.Parameters(
                QuantizationUnit.Parameters(tableType =
QuantizationUnit.TableType.CHROMATICITY(true))
            ),
            third: QuantizationManager.Parameters =
```

```
QuantizationManager.Parameters(
               QuantizationUnit.Parameters(tableType =
QuantizationUnit.TableType.CHROMATICITY(true))
           )
       ) : this (managerParams = {
           when(it) {
               0 -> first
               1 -> second
               2 -> third
               else -> throw Exception("can't hold it")
           }
       })
   }

   private val quantizationManagerMap = LazyMap<Int, QuantizationManager> {
       QuantizationManager(parameters.managerParams(it))
   }

   private val collectionProcessor by lazy {
       val directFun = { pair: Pair<Int, Matrix<PixelType>> ->
           pair.first to
quantizationManagerMap.getOrLazy(pair.first).direct(pair.second)
       }
       val reverseFun = { pair: Pair<Int, Matrix<PixelType>> ->
           pair.first to
quantizationManagerMap.getOrLazy(pair.first).reverse(pair.second)
       }
       return@lazy when (parameters.asyncType) {
           AsyncType.SINGLE ->
AsyncCollectionProcessor.SINGLE_THREAD(directFun, reverseFun)
           AsyncType.MULTI_THREAD ->
AsyncCollectionProcessor.MULTI_THREADS(3, directFun, reverseFun)
       }
   }

   override fun processDirectTyped(data: ProcessingData.Image):
ProcessingData.Image {
       val res = direct(ProcessingState(planesOfPixel =
data.triple)).planesOfPixel!!
       return ProcessingData.Image(Triple(res[0], res[1], res[2]))
   }

   override fun processReverseTyped(data: ProcessingData.Image):
ProcessingData.Image {
       val res = reverse(ProcessingState(planesOfPixel =
data.triple)).planesOfPixel!!
       return ProcessingData.Image(Triple(res[0], res[1], res[2]))
   }

   override fun direct(data: ProcessingState): ProcessingState {
       val pairs = data.planesOfPixel!!.mapIndexed { index: Int, ints:
Matrix<PixelType> ->
           quantizationManagerMap.getOrLazy(index).compressionTypeMatrix =
data.planesOfCompressionType?.get(index)
           index to ints
       }
       data.planesOfPixel = collectionProcessor.direct(pairs).map { it.second
}
       return data
   }

   override fun reverse(data: ProcessingState): ProcessingState {
       val pairs = data.planesOfPixel!!.mapIndexed { index: Int, ints:
Matrix<PixelType> ->
```

```
            quantizationManagerMap.getOrLazy(index).compressionTypeMatrix =
data.planesOfCompressionType?.get(index)
            index to ints
        }
        data.planesOfPixel = collectionProcessor.reverse(pairs).map {
it.second }
        return data
    }
}

class ModuleOpc3(
    val parameters: Parameters = Parameters()
) : IDataProcessor2<ProcessingState, ProcessingState> {

    data class Parameters(
        val asyncType: AsyncType = AsyncType.MULTI_THREAD,
        val managerParamProvider: (Int) -> Opc3ProcessingManager.Parameters =
{ Opc3ProcessingManager.Parameters() }
    )

    private val managerMap = LazyMap<Int, Opc3ProcessingManager> {
        Opc3ProcessingManager(parameters.managerParamProvider(it))
    }

    private val asyncCollectionProcessor by lazy {
        val directF = { pair: Pair<Int, Matrix<PixelType>> ->
            pair.first to managerMap.getOrLazy(pair.first).direct(pair.second)
        }
        val reverseF = { pair: Pair<Int, Matrix<DataOpc3>> ->
            pair.first to
managerMap.getOrLazy(pair.first).reverse(pair.second)
        }
        when (parameters.asyncType) {
            AsyncType.SINGLE ->
AsyncCollectionProcessor.SINGLE_THREAD(directF, reverseF)
            AsyncType.MULTI_THREAD ->
AsyncCollectionProcessor.MULTI_THREADS(3, directF, reverseF)
        }
    }

    override fun direct(data: ProcessingState) = data.also {
        val pairs = it.planesOfPixel!!.mapIndexed { index: Int, ints:
Matrix<PixelType> -> index to ints }
        it.planesOfSize = it.planesOfSize ?: it.planesOfPixel!!.map {
it.sizeTyped }
        it.planesOfDataOpc3 = asyncCollectionProcessor.direct(pairs).map {
it.second }
    }

    override fun reverse(data: ProcessingState) = data.also {
        val pairs = it.planesOfDataOpc3!!.mapIndexed { index: Int, ints:
Matrix<DataOpc3> -> index to ints }
        it.planesOfPixel = asyncCollectionProcessor.reverse(pairs).map {
it.second }
    }
}

class ModuleOpc3ToBits(
    val parameters: Parameters = Parameters()
) : IDataProcessor2<ProcessingState, ProcessingState> {

    data class Parameters(
        val saveFlags: Boolean = true,
        val saveBases: Boolean = true,
```

```
        val saveCode: Boolean = true,
        val saveSign: Boolean = true,
        val logPrefix: String = ""
    )

    private val flagManager =

Opc3FlagToBitsManager(Opc3FlagToBitsManager.Parameters(parameters.logPrefix,
parameters.saveBases))
    private val baseManager =
Opc3BasesToBitsManager(Opc3BasesToBitsManager.Parameters(parameters.logPrefix
))
    private val codeManager =
Opc3CodeToBitsManager(Opc3CodeToBitsManager.Parameters(parameters.logPrefix))
    private val signManager =
Opc3SignToBitsManager(Opc3SignToBitsManager.Parameters(parameters.logPrefix))

    override fun direct(data: ProcessingState) = data.also {
        val bitVector = data.bitVector ?: BitVector()
        if (parameters.saveFlags)
            flagManager.direct(bitVector, data.planesOfDataOpc3!!)
        if (parameters.saveBases)
            baseManager.direct(bitVector, data.planesOfDataOpc3!!)
        if (parameters.saveCode)
            codeManager.direct(bitVector, data.planesOfDataOpc3!!)
        if (parameters.saveSign)
            signManager.direct(bitVector, data.planesOfDataOpc3!!)
        data.bitVector = bitVector
    }

    override fun reverse(data: ProcessingState) = data.also {
        val reader = data.bitVector?.getReader()!!
        val planesOfDataOpc3 = if (parameters.saveFlags)
            flagManager.reverse(reader)
        else
            data.planesOfDataOpc3

        if (parameters.saveBases)
            baseManager.reverse(reader, planesOfDataOpc3!!)
        if (parameters.saveCode)
            codeManager.reverse(reader, planesOfDataOpc3!!)
        if (parameters.saveSign)
            signManager.reverse(reader, planesOfDataOpc3!!)

        data.planesOfDataOpc3 = planesOfDataOpc3
    }
}
```

Додакток В

Акт реалізації результатів науково-прикладних досліджень

ЗАТВЕРДЖУЮ
ТВО Головного інженера
ТОВ "Одеський авіаційний завод"
Микола ГОНЧАРЕНКО
"___"______12______ 2023 р.

А К Т

впровадження результатів науково-прикладних досліджень
Баранніка Дмитра Володимировича

Комісія у складі: голови головного технолога Ользі Никифоренко та членів комісії провідного інженера Володимира Кондратова, провідного інженера технолога Юрія Ведмідя, провідного інженера Анатолія Якимчука склала дійсний акт. Він стосується використання в процесі виконання дослідно-конструкторських робіт результатів науково-прикладних досліджень Баранніка Дмитра Володимировича. Сюди відносяться наступні:

1. Технологія диференційного стеганокомпресійного перетворення на основі позиційно-поліадичного кодування в стегано-поліадичному базисі. Основні компоненти технології стосуються врахування: рівня інформативності відеосегментів за структурно-семантичним змістом, а саме: здійснення СК-перетворень в позиційному диференціально-поліадичному та спектрально-адичному базисах; в процесі змішаного маскування відсутності нульових компонент спектральноадичного базису. Це дозволяє підвищити швидкість стеганокомпресійного каналу передачі інформації.

2. Метод змішаного стеганокомпресійного кодування на основі позиційних систем в ***маскованому*** поліадичному базисі. Основні технологічні компоненти методу стосуються створення стеганокомпресійного кодування в змішано-маскованому після форматування стегано-поліадичному базисі з реалізацією двокаскадної ***кодово-компресійної*** імплементації доданого до стегано-поліадичного числа елементу прихованого повідомлення.

3. Метод двокаскадної кодової рекомбінації прихованої інформації на основі зворотних стеганокомпресійних перетворень. Основа методу полягає у тому що реконструкція стегано-поліадичних чисел організується за двокаскадною концепцією, на основі масковано-залежного позиційного декодуван-

ня стегано-кодового значення в форматованому стегано-поліадичному базисі з попереднім вилученням з позиціонованої СК-кодограми за допомогою нерівномірного двійкового блокового кодування її змісту.

Впровадження результатів досліджень Баранніка Дмитра в системах надання відеосервісів з використанням дистанційних платформ збору, обробки та передачі прихованої інформації та бортових відеоінформаційних ресурсів дозволили забезпечити наступну перевагу відносно сучасних технологій в умовах виключення втрат цілісності для санкціоновано-вилученої прихованої інформації, а саме досягається:

1) одночасне збільшення: швидкості стеганокомпресійного каналу в середньому в 2,1 рази; рівня семантичної цілісності відновленої послідовності відеосегментів за показником пікового відношення сигнал/шум в середньому на 16,3 дБ; підвищення рівня маскування прихованої у ВСК інформації у разі спроби несанкціонованого доступу в середньому на 11 дБ;

2) створення умов для встановленої стеганографічної ємності щодо одночасної доставки в реальному часі:

- для швидкості каналу передачі даних 25 Мбіт/с- відео-контейнерів форматом не менш ніж формат 4К Ultra HDHV (3840x2160) та об'єму прихованої інформації на рівні 30 % від бітового об'єму відеокадру форматом Full HD (1920 x 1080)

- для швидкості каналу передачі даних 55 Мбіт/с - відео-контейнерів форматом не менш ніж 4К UHD (4128x3096) та об'єму прихованої інформації на рівні 10 % від бітового об'єму відеокадру форматом 4К UHD (3840x2160).

Голова комісії: Ольга НИКИФОРЕНКО

Члени комісії: Володимир КОНДРАТОВ

Юрій ВЕДМІДЬ

Анатолій ЯКИМЧУК

ЗАТВЕРДЖУЮ
Директор ТОВ «ЮЕЙ ДЕФЕНС»
Дмитро ЛЮЛІН
«19» 10 2023 р.

"ЮЕЙ ДЕФЕНС" Ідентифікаційний код 43521080

А К Т

впровадження результатів науково-прикладних досліджень
Баранніка Дмитра Володимировича

Комісія у складі: голови начальника відділу Ляшенко В.Г., та членів комісії Панченко С.М., Синовята А.М. Він стосується використання в процесі виконання дослідно-конструкторських робіт результатів наступних науково-прикладних досліджень Баранніка Дмитра Володимировича:

1. Технології локалізації структурної стеганографічної надмірності не пов'язаної з корекцією кодограми стеганокоду. Яка дозволяє здійснювати маскування структурної стеганографічної надмірності, що не пов'язане з корекції кодограми стеганокода;

2) Правила вбудовування інформації для структурного стеганографічного кодування, що полягає в тому, що: один біт приховуваного повідомлення вбудовується на другу позицію нерівновагового позиційного числа; модифікації піддається заснування першого елемента, шляхом збільшення його в два рази.

3. Структурно стенографічного кодування на основі імплантації біта приховуваного повідомлення на позицію другого елементу НПЧ з модифікацією нерівновагового позиційного базису.

Впровадження результатів досліджень Баранніка Дмитра в системах обробки та передачі прихованої інформації дозволили забезпечити наступну перевагу відносно сучасних технологій:

1) для розробленого стеганографічного методу ймовірність безпомилкового вилучення вбудованих даних в умовах відсутності атак на вбудоване

повідомлення дорівнює одиниці;

2) виграш для розробленого методу відносно методу наймолодшого біту за значенням ймовірності безпомилкового вилучення в умовах відсутності атак на вбудоване повідомлення складає 40%;

3) виграш для розробленого методу відносно існуючих стеганографічних технологій по величині стеганографічного бітрейту становить в середньому 25 %.

Голова комісії Ляшенко В.Г.

Члени комісії: Панченко С.М.
Синовята А.М.